

[illegible]

.....

RRRRRRRR	DDDDDDDD	BBBBBBBB	SSSSSSSS	HH	HH	RRRRRRRR				
RRRRRRRR	DDDDDDDD	BBBBBBBB	SSSSSSSS	HH	HH	RRRRRRRR				
RR	RR	DD	DD	BB	BB	SS	HH	HH	RR	RR
RR	RR	DD	DD	BB	BB	SS	HH	HH	RR	RR
RR	RR	DD	DD	BB	BB	SS	HH	HH	RR	RR
RR	RR	DD	DD	BB	BB	SS	HH	HH	RR	RR
RRRRRRRR	DD	DD	BBBBBBBB	SSSSSS	HHHHHHHHHH	RRRRRRRR				
RRRRRRRR	DD	DD	BBBBBBBB	SSSSSS	HHHHHHHHHH	RRRRRRRR				
RR	RR	DD	DD	BB	BB	SS	HH	HH	RR	RR
RR	RR	DD	DD	BB	BB	SS	HH	HH	RR	RR
RR	RR	DD	DD	BB	BB	SS	HH	HH	RR	RR
RR	RR	DD	DD	BB	BB	SS	HH	HH	RR	RR
RR	RR	DDDDDDDD	BBBBBBBB	SSSSSSSS	HH	HH	RR	RR	RR	RR
RR	RR	DDDDDDDD	BBBBBBBB	SSSSSSSS	HH	HH	RR	RR	RR	RR

```

LL          IIIIII          SSSSSSSS
LL          IIIIII          SSSSSSSS
LL          II             SS
LL          II             SS
LL          II             SS
LL          II             SS
LL          II             SSSSSS
LL          II             SSSSSS
LL          II             SS
LL          II             SS
LL          II             SS
LL          II             SS
LLLLLLLLLLLL IIIIII          SSSSSSSS
LLLLLLLLLLLL IIIIII          SSSSSSSS

```



```
0001 0 %TITLE 'RDBSHR - Rights database loadable system services'
0002 0 MODULE RDBSHR (IDENT = 'V04-000') =
0003 1 BEGIN
0004 1
0005 1
0006 1 *****
0007 1 *
0008 1 *   COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
0009 1 *   DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
0010 1 *   ALL RIGHTS RESERVED.
0011 1 *
0012 1 *   THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
0013 1 *   ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
0014 1 *   INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
0015 1 *   COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
0016 1 *   OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
0017 1 *   TRANSFERRED.
0018 1 *
0019 1 *   THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
0020 1 *   AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
0021 1 *   CORPORATION.
0022 1 *
0023 1 *   DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
0024 1 *   SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
0025 1 *
0026 1 *
0027 1 *****
0028 1
0029 1
0030 1 ++
0031 1 FACILITY: EXECUTIVE, SYSTEM SERVICES
0032 1
0033 1 ABSTRACT:
0034 1
0035 1   This module contains system services that maintain the rights
0036 1   database. It is built as a privileged shareable image. The
0037 1   remaining rights database system services are in the exec.
0038 1   The system services in this module are:
0039 1
0040 1           $ADD_HOLDER    $ADD_IDENT    $CREATE_RDB
0041 1           $FIND_HOLD    $FIND_HOLDER  $MOD_HOLDER
0042 1           $MOD_IDENT    $REM_HOLDER   $REM_IDENT
0043 1
0044 1 ENVIRONMENT:
0045 1
0046 1   VAX/VMS native mode, user, supervisor, or exec modes.
0047 1
0048 1 --
0049 1
0050 1 AUTHOR: Andrew C. Goldstein, CREATION DATE: 16-Nov-1982 18:51
0051 1
0052 1 MODIFIED BY:
0053 1
0054 1   V03-013 ACG0447      Andrew C. Goldstein,    23-Aug-1984 16:35
0055 1   Upcase all input identifier names
0056 1
0057 1   V03-012 JRL0009      John R. Lawson, Jr.     29-Jun-1984 11:28
```

```
58 0058 1 | Check for existence of current Rights Data Base.
59 0059 1 | Do not allow $CREATE if present.
60 0060 1 |
61 0061 1 | V03-011 LY0469 Larry Yetto 22-MAR-1984 14:13
62 0062 1 | Add two new parameters to the MOD_IDENT service to
63 0063 1 | allow the identifier name or value to be modified. This
64 0064 1 | change also requires the CHG attribut to be associated
65 0065 1 | with the secondary and tertiary keys therefore the create service
66 0066 1 | was also modified.
67 0067 1 |
68 0068 1 | V03-010 RSH0100 R. Scott Hanna 03-Feb-1984
69 0069 1 | Update comments to indicate that the ATTRIB parameter is
70 0070 1 | optional for $ADD_HOLDER AND $ADD_IDENT.
71 0071 1 |
72 0072 1 | V03-009 RSH0088 R. Scott Hanna 05-Jan-1984
73 0073 1 | Change $ADD_HOLDER, $ADD_IDENT, $MOD_HOLDER, $MOD_IDENT,
74 0074 1 | $REM_HOLDER, and $REM_IDENT to return $$$_NORMAL instead
75 0075 1 | of RM$$_NORMAL.
76 0076 1 |
77 0077 1 | V03-008 TMK0001 Todd M. Katz 22-Oct-1983
78 0078 1 | The name of the real system service entry point for
79 0079 1 | $FINISH_RDB has been changed from EXE$FINISH_RDB to
80 0080 1 | EXE$$$FINISH_RDB. This change was required because the
81 0081 1 | system service could no longer be reached directly from the
82 0082 1 | executive mode system service dispatcher.
83 0083 1 |
84 0084 1 | V03-007 RSH0062 R. Scott Hanna 12-Sep-1983
85 0085 1 | Modify ID value validation to return $$$_IDENT for
86 0086 1 | an ID of 0. (All services except $ADD_IDENT)
87 0087 1 |
88 0088 1 | V03-006 RSH0050 R. Scott Hanna 28-Jul-1983
89 0089 1 | Changed SY$$CREATE_RDB to use the new FAB bits
90 0090 1 | FAB$V_LNM_MODE instead of FAB$B_DSBMSK.
91 0091 1 |
92 0092 1 | V03-005 RSH0046 R. Scott Hanna 24-Jul-1983
93 0093 1 | Modified SY$$CREATE_RDB to create records for the
94 0094 1 | environmental rights.
95 0095 1 |
96 0096 1 | V03-004 RSH0041 R. Scott Hanna 21-Jun-1983
97 0097 1 | Provide additional ID name validation. Open the rights
98 0098 1 | database as a process permanent file when there is no
99 0099 1 | active image.
100 0100 1 |
101 0101 1 | V03-003 RSH0033 R. Scott Hanna 26-May-1983
102 0102 1 | Modify FAB in EXE$OPEN_RDB so that a logical name
103 0103 1 | can be used for the Rights Database.
104 0104 1 |
105 0105 1 | V03-002 GAS0126 Gerry Smith 26-May-1983
106 0106 1 | Put SY$$FIND_HELD into its own module. This is
107 0107 1 | necessary so that LOGINOUT can reference it before
108 0108 1 | the loadable system services are loaded, at boot time.
109 0109 1 |
110 0110 1 | V03-001 RSH0008 R. Scott Hanna 01-Mar-1983
111 0111 1 | Changed SY$$CREATE_RDB to call EXE$SET_RDIPT.
112 0112 1 |
113 0113 1 | **
114 0114 1 |
```



```
0115 1
0116 1 LIBRARY 'SYS$LIBRARY:LIB.L32';
0117 1
0118 1 FORWARD ROUTINE
0119 1     SYSSADD_HOLDER,
0120 1     SYSSADD_IDENT,
0121 1     SYSSCREATE_RDB,
0122 1     SYSSFIND_HOLDER,
0123 1     SYSSMOD_HOLDER,
0124 1     SYSSMOD_IDENT,
0125 1     SYSSMOD_IDENT_ATTRIB,
0126 1     SYSSMOD_IDENT_ID,
0127 1     SYSSMOD_IDENT_NAME,
0128 1     SYSSREM_HOLDER,
0129 1     SYSSREM_IDENT;
0130 1
0131 1 LINKAGE
0132 1     EXE_VAL_IDNAME = JSB (REGISTER=1; REGISTER=1, REGISTER=2) :
0133 1                     NOPRESERVE (3)
0134 1                     NOTUSED (4,5,6,7,8,9,10,11),
0135 1     EXE_ALOP1IMAG = JSB (REGISTER=1; REGISTER=1, REGISTER=2) :
0136 1                     NOPRESERVE (3);
0137 1
0138 1 EXTERNAL ROUTINE
0139 1     EXESOPEN_RDB : ADDRESSING_MODE (ABSOLUTE),
0140 1     EXESCLOSE_RDB : NOVALUE ADDRESSING_MODE (ABSOLUTE),
0141 1     EXES$FINISH_RDB : ADDRESSING_MODE (ABSOLUTE),
0142 1     EXESALOP1IMAG : EXE_ALOP1IMAG ADDRESSING_MODE (ABSOLUTE),
0143 1     EXESVAL_IDNAME : EXE_VAL_IDNAME ADDRESSING_MODE (ABSOLUTE),
0144 1     EXESSET_RDIPTR : ADDRESSING_MODE (ABSOLUTE),
0145 1     SYSSCMKRNL : ADDRESSING_MODE (ABSOLUTE);
0146 1
0147 1 EXTERNAL
0148 1     CTL$GL_RDIPTR : REF $BBLOCK ADDRESSING_MODE (ABSOLUTE),
0149 1     CTL$GL_IMGHDRBF : LONG ADDRESSING_MODE (ABSOLUTE),
0150 1     EXEST_ID_UPCASE : VECTOR [,BYTE] ADDRESSING_MODE (GENERAL);
0151 1
0152 1 BUILTIN
0153 1     PROBER,
0154 1     PROBEW;
0155 1
0156 1
0157 1 LITERAL
0158 1     UIC$M_ID_FORM_FLAG = 1*31, ! mask for id form of identifier
0159 1     KGB$M_VACID_ATTRIB = KGB$M_RESOURCE; ! mask of valid attributes
```



```
161 0160 1 %SBTTL ' SYSSADD HOLDER - add holder to RDB'
162 0161 1 GLOBAL ROUTINE SYSSADD_HOLDER (ID, HOLDER, ATTRIB) =
163 0162 1
164 0163 1 ++
165 0164 1
166 0165 1 FUNCTIONAL DESCRIPTION:
167 0166 1
168 0167 1 This routine adds the specified holder record to the rights
169 0168 1 database.
170 0169 1
171 0170 1 CALLING SEQUENCE:
172 0171 1 SYSSADD_HOLDER (ID, HOLDER, ATTRIB)
173 0172 1
174 0173 1 INPUT PARAMETERS:
175 0174 1 ID: identifier longword to associate the
176 0175 1 holder record with
177 0176 1 HOLDER: address of the holder identifier quadword
178 0177 1 ATTRIB: (optional) attributes longword to grant to the holder
179 0178 1
180 0179 1 IMPLICIT INPUTS:
181 0180 1 NONE
182 0181 1
183 0182 1 OUTPUT PARAMETERS:
184 0183 1 NONE
185 0184 1
186 0185 1 IMPLICIT OUTPUTS:
187 0186 1 NONE
188 0187 1
189 0188 1 ROUTINE VALUE:
190 0189 1 Status of operation
191 0190 1
192 0191 1 SIDE EFFECTS:
193 0192 1 Holder record created
194 0193 1
195 0194 1 --
196 0195 1
197 0196 2 BEGIN
198 0197 2
199 0198 2 LOCAL
200 0199 2 LOC_ID : LONG, ! local copy of ID
201 0200 2 LOC_HOLDER : REF VECTOR, ! local copy of HOLDER
202 0201 2 HOLDER_ID : VECTOR [2], ! local copy of holder id quadword
203 0202 2 LOC_ATTRIB : LONG, ! local copy of ATTRIB
204 0203 2 ID_ATTRIB : LONG, ! attributes of identifier
205 0204 2 STATUS : LONG, ! general status value
206 0205 2 CLOSE : LONG, ! call to EXE$CLOSE_RDB required flag
207 0206 2 RAB : $RAB DECL, ! RAB for file operations
208 0207 2 REC_BUFFER : $BBLOCK [KGB$K_IDENT_RECORD]; ! general purpose record buffer
209 0208 2
210 0209 2
211 0210 2 LABEL
212 0211 2 RDB_OPEN; ! rights database is open in this block
213 0212 2
214 0213 2 ! Validate parameters
215 0214 2
216 0215 2
217 0216 2 LOC_ID = .ID;
```

```
218 0217 2 IF (.LOC_ID AND UIC$M_ID_FORM_FLAG) NEQU 0
219 0218 2 THEN
220 0219 2 (IF (.LOC_ID GTRU UIC$K_LAST_ID) THEN RETURN SS$_IVIDENT)
221 0220 2 ELSE
222 0221 2 (IF (.LOC_ID GTRU UIC$K_MAX_UIC) OR (.LOC_ID EQL 0) THEN RETURN SS$_IVIDENT);
223 0222 2
224 0223 2 LOC HOLDER = .HOLDER;
225 0224 2 IF NOT PROBER (%REF(0), %REF(8), .LOC_HOLDER) THEN RETURN SS$_ACCVIO;
226 0225 2 HOLDER_ID[0] = .LOC_HOLDER[0];
227 0226 2 HOLDER_ID[1] = .LOC_HOLDER[1];
228 0227 2 IF .HOLDER_ID[0] GTRU UIC$K_MAX_UIC OR
229 0228 2 .HOLDER_ID[0] EQLU .LOC_ID OR
230 0229 2 .HOLDER_ID[1] NEQU 0
231 0230 2 THEN
232 0231 2 RETURN SS$_IVIDENT;
233 0232 2
234 0233 2 LOC_ATTRIB = .ATTRIB;
235 0234 2 IF 7.LOC_ATTRIB AND NOT KGB$M_VALID_ATTRIB) NEQU 0 THEN RETURN SS$_BADPARAM;
236 0235 2
237 0236 2 ! Get the rights database open for write.
238 0237 2 !
239 0238 2
240 P 0239 2 $RAB_INIT (RAB = RAB,
241 P 0240 2 RAC = KEY,
242 P 0241 2 KRF = 0,
243 P 0242 2 KBF = HOLDER_ID[0],
244 P 0243 2 KSZ = 4,
245 P 0244 2 ROP = (NLK, RRL),
246 P 0245 2 UBF = REC_BUFFER,
247 P 0246 2 USZ = KGB$K_IDENT_RECORD
248 0247 2 );
249 0248 2 STATUS = EXE$OPEN RDB (0, 1, RAB[RAB$W_ISI], CLOSE);
250 0249 2 IF NOT .STATUS THEN RETURN .STATUS;
251 0250 2
252 0251 2 RDB_OPEN:
253 0252 2 BEGIN
254 0253 2
255 0254 2 ! Check to make sure that the holder ID exists.
256 0255 2 !
257 0256 2
258 0257 2 STATUS = $FIND (RAB = RAB);
259 0258 2 IF .STATUS EQLU RMSS$_RNF THEN STATUS = SS$_NOSUCHID;
260 0259 2 IF NOT .STATUS THEN LEAVE RDB_OPEN;
261 0260 2
262 0261 2 ! Read and lock the ident record and save away its attributes.
263 0262 2 !
264 0263 2
265 0264 2 RAB[RAB$V_RRL] = 0;
266 0265 2 RAB[RAB$V_NLK] = 0;
267 0266 2 RAB[RAB$V_RLK] = 1;
268 0267 2 RAB[RAB$V_ULK] = 1;
269 0268 2 RAB[RAB$V_WAT] = 1;
270 0269 2 RAB[RAB$L_KBF] = LOC_ID;
271 0270 2 STATUS = $GET (RAB = RAB);
272 0271 2 IF .STATUS EQLU RMSS$_RNF THEN STATUS = SS$_NOSUCHID;
273 0272 2 IF NOT .STATUS
274 0273 2 THEN
```



```
275 0274 4 BEGIN
276 0275 4 $FREE (RAB = RAB);
277 0276 4 LEAVE RDB_OPEN;
278 0277 4 END;
279 0278 3 ID_ATTRIB = .REC_BUFFER[KGB$L_ATTRIBUTES];
280 0279 3
281 0280 3 ! Now read all holder records to make sure that the specified holder
282 0281 3 ! doesn't already exist.
283 0282 3 !
284 0283 3
285 0284 3 RAB[RAB$V_RLK] = 0;
286 0285 3 RAB[RAB$V_ULK] = 0;
287 0286 3 RAB[RAB$V_WAT] = 0;
288 0287 3 RAB[RAB$V_RRL] = 1;
289 0288 3 RAB[RAB$V_NLK] = 1;
290 0289 3 RAB[RAB$V_LIM] = 1;
291 0290 3 RAB[RAB$B_RAC] = RAB$C_SEQ;
292 0291 3 WHILE 1 DO
293 0292 4 BEGIN
294 0293 4 STATUS = $GET (RAB = RAB);
295 0294 4 IF .STATUS EQLU RMSS_EOF OR .STATUS EQLU RMSS_OK_LIM THEN EXITLOOP;
296 0295 4 IF NOT .STATUS
297 0296 4 THEN
298 0297 5 BEGIN
299 0298 5 $FREE (RAB = RAB);
300 0299 5 LEAVE RDB_OPEN;
301 0300 4 END;
302 0301 4 IF CH$EQL (KGB$$_HOLDER, HOLDER_ID[0], KGB$$_HOLDER, REC_BUFFER[KGB$Q_HOLDER])
303 0302 4 THEN
304 0303 5 BEGIN
305 0304 5 STATUS = SS$_DUPIDENT;
306 0305 5 $FREE (RAB = RAB);
307 0306 5 LEAVE RDB_OPEN;
308 0307 4 END;
309 0308 3 END;
310 0309 3
311 0310 3 ! Finally build and write the new holder record.
312 0311 3 !
313 0312 3
314 0313 3 RAB[RAB$B_RAC] = RAB$C_KEY;
315 0314 3 RAB[RAB$W_RSZ] = KGB$K_HOLD_RECORD;
316 0315 3 RAB[RAB$L_RBF] = REC_BUFFER;
317 0316 3 REC_BUFFER[KGB$L_IDENTIFIER] = .LOC_ID;
318 0317 3 REC_BUFFER[KGB$L_ATTRIBUTES] = .LOC_ATTRIB AND .ID_ATTRIB;
319 0318 3 CH$MOVE (KGB$$_HOLDER, HOLDER_ID[0], REC_BUFFER[KGB$Q_HOLDER]);
320 0319 3 STATUS = $PUT (RAB = RAB);
321 0320 3 $FREE (RAB = RAB);
322 0321 3 END;
323 0322 2
324 0323 2 ! Close the rights database if there is no image
325 0324 2 !
326 0325 2
327 0326 2 IF .CLOSE THEN EXE$CLOSE_RDB();
328 0327 2 IF .STATUS
329 0328 2 THEN
330 0329 2 RETURN SS$_NORMAL
331 0330 2 ELSE
```


RDBSHR
V04-000

RDBSHR - Rights database loadable system service
SYSS\$ADD_HOLDER - add holder to RDB

C 8
16-Sep-1984 01:48:50
14-Sep-1984 12:40:52

VAX-11 Bliss-32 V4.0-742
[LOADSS.SRC]RDBSHR.B32;1

Page 7
(2)

: 332
: 333
: 334
0331 2 RETURN .STATUS;
0332 2
0333 1 END;

! End of routine SYSS\$ADD_HOLDER

.TITLE RDBSHR RDBSHR - Rights database loadable system
service
.IDENT \V04-000\

.EXTRN EXE\$OPEN_RDB, EXE\$CLOSE_RDB
.EXTRN EXE\$FINISH_RDB
.EXTRN EXE\$ALOP1IMAG, EXE\$VAL_IDNAME
.EXTRN EXE\$SET_RDIPT, SYSS\$CMRNL
.EXTRN CTL\$GL_RDIPT, CTL\$GL_IMGHDRBF
.EXTRN EXE\$ID_UPCASE
.EXTRN SYSS\$FIND, SYSS\$GET
.EXTRN SYSS\$FREE, SYSS\$PUT

.PSECT \$CODE\$,NOWRT,2

.ENTRY SYSS\$ADD_HOLDER, Save R2,R3,R4,R5,R6,R7,R8,- R9 ; 0161

MOVAB SYSS\$GET, R9
MOVAB -132(SP), SP
MOVL ID, LOC_ID ; 0216
MOVL LOC_ID, R7 ; 0217
BGEQ 1\$
CMPL R7, #-1879048193 ; 0219
BLEQU 2\$
BRB 4\$
CMPL R7, #1073741823 ; 0221
BGTRU 4\$
TSTL R7
BEQL 4\$
MOVL HOLDER, LOC_HOLDER ; 0223
PROBER #0, #8, (LOC_HOLDER) ; 0224
BNEQ 3\$
MOVL #12, R0
RET
MOVL (LOC_HOLDER), HOLDER_ID ; 0225
MOVL 4(LOC_HOLDER), HOLDER_ID+4 ; 0226
CMPL HOLDER_ID, #1073741823 ; 0227
BGTRU 4\$
CMPL HOLDER_ID, R7 ; 0228
BEQL 4\$
TSTL HOLDER_ID+4 ; 0229
BEQL 5\$
MOVZWL #8740, R0 ; 0231
RET
MOVL ATTRIB, LOC_ATTRIB ; 0233
BITL LOC_ATTRIB, #-2 ; 0234
BEQL 6\$
MOVL #20, R0
RET
MOVCS #0, (SP), #0, #68, \$RMS_PTR ; 0247
MOVW #17409, \$RMS_PTR

03FC 00000
59 00000000G 00 9E 00002
5E FF7C CE 9E 00009
04 AE 04 AC D0 0000E
57 04 AE D0 00013
0B 18 00017
8FFFFFFF 8F 57 D1 00019
0F 1B 00020
39 11 00022
3FFFFFFF 8F 57 D1 00024 1\$:
30 1A 0002B
57 D5 0002D
2C 13 0002F
60 50 06 AC D0 00031 2\$:
08 00 0C 00035
04 12 00039
50 0C D0 0003B
04 0003E
7C AE 60 D0 0003F 3\$:
FC AD 04 A0 D0 00043
3FFFFFFF 8F 7C AE D1 00048
0B 1A 00050
57 7C AE D1 00052
05 13 00056
FC AD D5 00058
06 13 0005B
50 2224 8F 3C 0005D 4\$:
04 00062
58 0C AC D0 00063 5\$:
FFFFFFFE 8F 58 D3 00067
04 13 0006E
50 14 D0 00070
04 00073
0044 8F 00 6E 00 2C 00074 6\$:
38 AE 38 AE 0007B
4401 8F B0 0007D

3C	AE	00100008	8F	DO	00083	MOVL	#1048584, \$RMS_PTR+4	
56	AE		01	90	0008B	MOVB	#1, \$RMS_PTR+30	
58	AE		30	B0	0008F	MOVW	#48, \$RMS_PTR+32	
5C	AE	08	AE	9E	00093	MOVAB	REC_BUFFER, \$RMS_PTR+36	
68	AE	7C	AE	9E	00098	MOVAB	HOLDER_ID, \$RMS_PTR+48	
6C	AE		04	90	0009D	MOVB	#4, \$RMS_PTR+52	
			5E	DD	000A1	PUSHL	SP	0248
		3E	AE	9F	000A3	PUSHAB	RAB+2	
			01	DD	000A6	PUSHL	#1	
			7E	D4	000A8	CLRL	-(SP)	
00000000G	9F		04	FB	000AA	CALLS	#4, @#EXE\$OPEN_RDB	
	56		50	DO	000B1	MOVL	R0, STATUS	
	03		56	EB	000B4	BLBS	STATUS, 7\$	0249
		00D3	31	000B7	BRW	16\$		
		38	AE	9F	000BA	PUSHAB	RAB	0257
00000000G	00		01	FB	000BD	CALLS	#1, SYSS\$FIND	
	56		50	DO	000C4	MOVL	R0, STATUS	
000182B2	8F		56	D1	000C7	CMPL	STATUS, #98994	0258
			05	12	000CE	BNEQ	8\$	
	56	21EC	8F	3C	000D0	MOVZWL	#8684, STATUS	
	03		56	EB	000D5	BLBS	STATUS, 9\$	0259
			00A1	31	000D8	BRW	14\$	
3C	AE	00100008	8F	CA	000DB	BICL2	#1048584, RAB+6	0265
3E	AE		0E	88	000E3	BISB2	#14, RAB+6	0268
68	AE	04	AE	9E	000E7	MOVAB	LOC_ID, RAB+48	0269
		38	AE	9F	000EC	PUSHAB	RAB	0270
	69		01	FB	000EF	CALLS	#1, SYSS\$GET	
	56		50	DO	000F2	MOVL	R0, STATUS	
000182B2	8F		56	D1	000F5	CMPL	STATUS, #98994	0271
			05	12	000FC	BNEQ	10\$	
	56	21EC	8F	3C	000FE	MOVZWL	#8684, STATUS	
	6C		56	E9	00103	BLBC	STATUS, 13\$	0272
	54	0C	AE	DO	00106	MOVL	REC_BUFFER+4, ID_ATTRIB	0278
3E	AE		0E	8A	0010A	BICB2	#14, RAB+6	0286
3C	AE	00104008	8F	C8	0010E	BISL2	#1064968, RAB+5	0289
			AE	94	00116	CLRB	RAB+30	0290
		38	AE	9F	00119	PUSHAB	RAB	0293
	69		01	FB	0011C	CALLS	#1, SYSS\$GET	
	56		50	DO	0011F	MOVL	R0, STATUS	
0001827A	8F		56	D1	00122	CMPL	STATUS, #98938	0294
			1B	13	00129	BEQL	12\$	
00018051	8F		56	D1	0012B	CMPL	STATUS, #98385	
			12	13	00132	BEQL	12\$	
			56	E9	00134	BLBC	STATUS, 13\$	0295
10	AE	7C	AE	08	29	CMPC3	#8, HOLDER_ID, REC_BUFFER+8	0301
			DA	12	0013D	BNEQ	11\$	
		56	8F	3C	0013F	MOVZWL	#8748, STATUS	0304
			2C	11	00144	BRB	13\$	0305
	56	AE	01	90	00146	MOVB	#1, RAB+30	0313
	5A	AE	10	B0	0014A	MOVW	#16, RAB+34	0314
	60	AE	08	AE	9E	MOVAB	REC_BUFFER, RAB+40	0315
	08	AE	57	DO	00153	MOVL	R7, REC_BUFFER	0316
			54	D2	00157	MCOML	ID_ATTRIB, R0	0317
0C	AE		50	CB	0015A	BICL3	R0, LOC_ATTRIB, REC_BUFFER+4	
10	AE	7C	AE	08	28	MOV3	#8, HOLDER_ID, REC_BUFFER+8	0318
			AE	9F	00165	PUSHAB	RAB	0319
00000000G	00		01	FB	00168	CALLS	#1, SYSS\$PUT	

RDBSHR
V04-000

RDBSHR - Rights database loadable system servic
SYSS\$ADD_HOLDER - add holder to RDB

E 8
16-Sep-1984 01:48:50
14-Sep-1984 12:40:52

VAX-11 Bliss-32 V4.0-742
[LOADSS.SRC]RDBSHR.B32;1

Page 9
(2)

	56		50	D0	0016F		MOVL	R0, STATUS	
		38	AE	9F	00172	13\$:	PUSHAB	RAB	
00000000G	00		01	FB	00175		CALLS	#1, SYSS\$FREE	0320
	07		6E	E9	0017C	14\$:	BLBC	CLOSE, 15\$	0326
00000000G	9F		00	FB	0017F		CALLS	#0, @#EXE\$CLOSE_RDB	
	04		56	E9	00186	15\$:	BLBC	STATUS, 16\$	0327
	50		01	D0	00189		MOVL	#1, R0	0331
				04	0018C		RET		
	50		56	D0	0018D	16\$:	MOVL	STATUS, R0	
				04	00190		RET		0333

; Routine Size: 401 bytes, Routine Base: \$CODE\$ + 0000


```
0334 1 %SBTTL ' SYSSADD_IDENT - add identifier to RDB'
0335 1 GLOBAL ROUTINE SYSSADD_IDENT (NAME, ID, ATTRIB, RESID) =
0336 1
0337 1 ++
0338 1
0339 1 FUNCTIONAL DESCRIPTION:
0340 1
0341 1     This routine creates the identifier of the specified name.
0342 1     If an explicit identifier code is given, it is used; otherwise
0343 1     the next available general code is used.
0344 1
0345 1 CALLING SEQUENCE:
0346 1     SYSSADD_IDENT (NAME, ID, ATTRIB, RESID)
0347 1
0348 1 INPUT PARAMETERS:
0349 1     NAME:    address of the identifier name character
0350 1             string descriptor
0351 1     ID:      (optional) identifier longword to associate with 'name'
0352 1     ATTRIB:  (optional) attributes longword to grant to the identifier
0353 1
0354 1 IMPLICIT INPUTS:
0355 1     NONE
0356 1
0357 1 OUTPUT PARAMETERS:
0358 1     RESID:   (optional) address of a longword to return the assigned
0359 1             identifier
0360 1
0361 1 IMPLICIT OUTPUTS:
0362 1     NONE
0363 1
0364 1 ROUTINE VALUE:
0365 1     success or failure status
0366 1
0367 1 SIDE EFFECTS:
0368 1     identifier record created
0369 1
0370 1 --
0371 1
0372 2 BEGIN
0373 2
0374 2 LOCAL
0375 2     LOC_NAME      : REF VECTOR,      ! local copy of NAME
0376 2     LENGTH        : LONG,           ! output from EXESVAL_IDNAME
0377 2     ADDRESS        : LONG,           ! output from EXESVAL_IDNAME
0378 2     LOC_ID         : LONG,           ! local copy of ID
0379 2     IDENTIFIER     : LONG,           ! identifier code to use
0380 2     LOC_ATTRIB     : LONG,           ! local copy of ATTRIB
0381 2     LOC_RESID      : LONG,           ! local copy of RESID
0382 2     STATUS         : LONG,           ! general status value
0383 2     CLOSE          : LONG,           ! call to EXESCLOSE_RDB required flag
0384 2     RAB            : $RAB_DECL,      ! RAB for record operations
0385 2     MAINT_RFA      : $BBLOCK [RAB$$, RFA], ! RFA of maintenance record
0386 2     REC_BUFFER     : $BBLOCK [KGB$$, MAINT_RECORD], ! general record buffer
0387 2     NAME_BUFFER    : $BBLOCK [KGB$$, NAME], ! name key buffer
0388 2
0389 2
0390 2
0391 2
0392 2
```

```

393 0391 2
394 0392 2 LABEL
395 0393 2 RDB_OPEN;
396 0394 2 ! rights database is open in this block
397 0395 2 ! Validate parameters
398 0396 2 !
399 0397 2
400 0398 2 LOC_NAME = .NAME;
401 0399 2 STATUS = EXESVAL IDNAME( .LOC_NAME; LENGTH, ADDRESS);
402 0400 2 IF NOT .STATUS THEN RETURN .STATUS;
403 0401 2 CH$TRANSLATE (EXEST_ID_UPCASE, .LENGTH, .ADDRESS, ' ', KGB$S_NAME, NAME_BUFFER);
404 0402 2
405 0403 2 LOC_ID = .ID;
406 0404 2 IF (.LOC_ID AND UIC$M_ID_FORM_FLAG) NEQU 0
407 0405 2 THEN
408 0406 2 (IF (.LOC_ID GTRU UIC$K_LAST_ID) THEN RETURN SSS_IVIDENT)
409 0407 2 ELSE
410 0408 2 (IF (.LOC_ID GTRU UIC$K_MAX_UIC) THEN RETURN SSS_IVIDENT);
411 0409 2
412 0410 2 LOC_ATTRIB = .ATTRIB;
413 0411 2 IF (.LOC_ATTRIB AND NOT KGB$M_VALID_ATTRIB) NEQU 0 THEN RETURN SSS_BADPARAM;
414 0412 2
415 0413 2 LOC_RESID = .RESID;
416 0414 2 IF .LOC_RESID NEQU 0 AND NOT PROBEW (%REF(0), %REF(4), .LOC_RESID)
417 0415 2 THEN
418 0416 2 RETURN SSS_ACCVIO;
419 0417 2
420 0418 2 ! Get the rights database open for write.
421 0419 2 !
422 0420 2
P 0421 2 $RAB_INIT (RAB = RAB,
P 0422 2 RAC = KEY,
P 0423 2 KRF = 0,
P 0424 2 KSZ = 4,
P 0425 2 KBF = UPLIT (0),
P 0426 2 ROP = (WAT, RLK, ULK),
P 0427 2 USZ = KGB$K_MAINT_RECORD,
P 0428 2 UBF = REC_BUFFER
431 0429 2 );
432 0430 2 STATUS = EXESOPEN RDB (0, 1, RAB[RAB$W_ISI], CLOSE);
433 0431 2 IF NOT .STATUS THEN RETURN .STATUS;
434 0432 2
435 0433 2 RDB_OPEN:
436 0434 2 BEGIN
437 0435 2
438 0436 2 ! First read the maintenance record to interlock the entire operation.
439 0437 2 !
440 0438 2
441 0439 2 STATUS = $GET (RAB = RAB);
442 0440 2 IF NOT .STATUS
443 0441 2 THEN
444 0442 2 BEGIN
445 0443 2 $FREE (RAB = RAB);
446 0444 2 LEAVE RDB_OPEN;
447 0445 2 END;
448 0446 2 CH$MOVE (RAB$S_RFA, RAB[RAB$W_RFA], MAINT_RFA);
449 0447 2
```



```

: 450      0448      ! Now see if the specified name is already in use.
: 451      0449      !
: 452      0450
: 453      0451      RAB[RAB$V_WAT] = 0;
: 454      0452      RAB[RAB$V_ULK] = 0;
: 455      0453      RAB[RAB$V_RLK] = 0;
: 456      0454      RAB[RAB$V_NLK] = 1;
: 457      0455      RAB[RAB$V_RRL] = 1;
: 458      0456      RAB[RAB$B_KRF] = 2;
: 459      0457      RAB[RAB$B_KSZ] = KGB$S_NAME;
: 460      0458      RAB[RAB$L_KBF] = NAME_BUFFER;
: 461      0459      STATUS = $FIND (RAB = RAB);
: 462      0460      IF .STATUS THEN STATUS = SSS_DUPLNAM;
: 463      0461      IF .STATUS NEQU RMSS_RNF
: 464      0462      THEN
: 465      0463      BEGIN
: 466      0464      $FREE (RAB = RAB);
: 467      0465      LEAVE RDB_OPEN;
: 468      0466      END;
: 469      0467
: 470      0468      ! If an explicit identifier is given, see if it is in use.
: 471      0469      !
: 472      0470
: 473      0471      RAB[RAB$B_KRF] = 0;
: 474      0472      RAB[RAB$B_KSZ] = 4;
: 475      0473
: 476      0474      IF .LOC_ID NEQU 0
: 477      0475      THEN
: 478      0476      BEGIN
: 479      0477      RAB[RAB$L_KBF] = LOC_ID;
: 480      0478      STATUS = $FIND (RAB = RAB);
: 481      0479      IF .STATUS THEN STATUS = SSS_DUPIDENT;
: 482      0480      IF .STATUS NEQU RMSS_RNF
: 483      0481      THEN
: 484      0482      BEGIN
: 485      0483      $FREE (RAB = RAB);
: 486      0484      LEAVE RDB_OPEN;
: 487      0485      END;
: 488      0486      IDENTIFIER = .LOC_ID;
: 489      0487      END
: 490      0488
: 491      0489      ! Otherwise we have to select an identifier.
: 492      0490      !
: 493      0491
: 494      0492      ELSE
: 495      0493      BEGIN
: 496      0494      IDENTIFIER = .REC_BUFFER[KGB$L_NEXT_ID];
: 497      0495
: 498      0496      ! Attempt to get the record pointed to by the new identifier. If it
: 499      0497      ! exists, keep incrementing until a free identifier is found. Wrap
: 500      0498      ! the identifier value when it overflows.
: 501      0499      !
: 502      0500
: 503      0501      RAB[RAB$L_KBF] = IDENTIFIER;
: 504      0502      WHILE 1 DO
: 505      0503      BEGIN
: 506      0504      IF .IDENTIFIER GTRU UIC$K_LAST_ID
```



```
507 0505 5 THEN
508 0506 5 IDENTIFIER = UIC$K FIRST_ID;
509 0507 5 STATUS = $FIND (RAB = RAB);
510 0508 5 IF NOT .STATUS
511 0509 5 THEN
512 0510 6 BEGIN
513 0511 6 IF .STATUS EQLU RMSS_RNF
514 0512 6 THEN
515 0513 6 EXITLOOP
516 0514 6 ELSE
517 0515 7 BEGIN
518 0516 7 $FREE (RAB = RAB);
519 0517 7 LEAVE RDB_OPEN;
520 0518 6 END;
521 0519 5 END;
522 0520 5 IDENTIFIER = .IDENTIFIER + 1;
523 0521 4 END;
524 0522 4
525 0523 4 ! Write back the maintenance record with an updated next identifier
526 0524 4 ! value. Note that while we increment the identifier here, it is
527 0525 4 ! not necessary to wrap it, since that is done in the check above.
528 0526 4 !
529 0527 4
530 0528 4 REC_BUFFER[KGB$ NEXT_ID] = .IDENTIFIER + 1;
531 0529 4 RAB[RAB$B RAC] = RAB$C RFA;
532 0530 4 CH$MOVE (RAB$S RFA, MAINT RFA, RAB[RAB$W_RFA]);
533 0531 4 STATUS = $FIND (RAB = RAB);
534 0532 4 IF NOT .STATUS
535 0533 4 THEN
536 0534 5 BEGIN
537 0535 5 $FREE (RAB = RAB);
538 0536 5 LEAVE RDB_OPEN;
539 0537 4 END;
540 0538 4
541 0539 4 STATUS = $UPDATE (RAB = RAB);
542 0540 4 IF NOT .STATUS
543 0541 4 THEN
544 0542 5 BEGIN
545 0543 5 $FREE (RAB = RAB);
546 0544 5 LEAVE RDB_OPEN;
547 0545 4 END;
548 0546 3 END;
549 0547 3
550 0548 3 IF .LOC_RESID NEQU 0 THEN .LOC_RESID = .IDENTIFIER;
551 0549 3
552 0550 3 ! Finally create the new identifier record and write it.
553 0551 3 !
554 0552 3
555 0553 3 REC_BUFFER[KGB$ IDENTIFIER] = .IDENTIFIER;
556 0554 3 REC_BUFFER[KGB$ ATTRIBUTES] = .LOC ATTRIB;
557 0555 3 CH$FILL (0, KGB$S HOLDER, REC_BUFFER[KGB$Q HOLDER]);
558 0556 3 CH$MOVE (KGB$S NAME, NAME_BUFFER, REC_BUFFER[KGB$T_NAME]);
559 0557 3 RAB[RAB$B RAC] = RAB$C KEY;
560 0558 3 RAB[RAB$W_RSZ] = KGB$K IDENT_RECORD;
561 0559 3 RAB[RAB$L RBF] = REC_BUFFER;
562 0560 3 STATUS = $PUT (RAB = RAB);
563 0561 3 $FREE (RAB = RAB);
```

```

: 564 0562 2      END;
: 565 0563 2
: 566 0564 2      ! Close the rights database if there is no image
: 567 0565 2      !
: 568 0566 2
: 569 0567 2      IF .CLOSE THEN EX$CLOSE_RDB();
: 570 0568 2      IF .STATUS
: 571 0569 2      THEN
: 572 0570 2          RETURN SS$_NORMAL
: 573 0571 2      ELSE
: 574 0572 2          RETURN .STATUS;
: 575 0573 2
: 576 0574 1      END;
```

! End of routine SYSSADD_IDENT

```

                                .PSECT $SPLIT$,NOWRT,NOEXE,2
                                00000000 00000 P.AAA: .LONG 0
                                .EXTRN SYSSUPDATE
                                .PSECT $CODE$,NOWRT,2
                                OFFC 00000
                                .ENTRY SYSSADD_IDENT, Save R2,R3,R4,R5,R6,R7,R8,-
                                R9,R10,R11
                                MOVAB SYSS$FIND, R11
                                MOVAB -184(SP), SP
                                MOVL NAME, LOC_NAME
                                JSB @#EX$VAL_IDNAME
                                MOVL R0, STATUS
                                BLBS STATUS, 1$
                                BRW 21$
                                MOVTC LENGTH, (ADDRESS), #32, EXEST_ID_UPCASE, -
                                #32, NAME_BUFFER
                                MOVL ID, LOC_ID
                                MOVL LOC_ID, R7
                                BGEQ 2$
                                CMPL R7, #-1879048193
                                BRB 3$
                                CMPL R7, #1073741823
                                BLEQU 4$
                                MOVZWL #8740, R0
                                RET
                                MOVL ATTRIB, LOC_ATTRIB
                                BITL LOC_ATTRIB, #-2
                                BEQL 5$
                                MOVL #20, R0
                                RET
                                MOVL RESID, LOC_RESID
                                CLRL R10
                                TSTL LOC_RESID
                                BEQL 6$
                                INCL R10
                                PROBEW #0, #4, (LOC_RESID)
                                BNEQ 6$
                                MOVL #12, R0
                                00000000G 00      20      0C      04
                                5B 00000000G 00 9E 00002
                                5E FF48 CE 9E 00009
                                51 04 AC D0 0000E
                                00000000G 9F 16 00012
                                56 50 D0 00018
                                03 56 E8 0001B
                                01CA 31 0001E
                                62 51 2E 00021 1$:
                                0C AE 20 0002A
                                04 AE 08 AC D0 0002D
                                57 04 AE D0 00032
                                09 18 00036
                                8F 57 D1 00038
                                8F 07 11 0003F
                                8F 57 D1 00041 2$:
                                3F 06 1B 00048 3$:
                                50 2224 8F 3C 0004A
                                04 04 0004F
                                59 0C AC D0 00050 4$:
                                FFF FFFE 8F 59 D3 00054
                                04 13 0005B
                                50 14 D0 0005D
                                04 00060
                                58 10 AC D0 00061 5$:
                                5A D4 00065
                                58 D5 00067
                                0C 13 00069
                                5A D6 0006B
                                68 04 00 0D 0006D
                                04 12 00071
                                50 0C D0 00073
                                0335
                                0398
                                0399
                                0400
                                0401
                                0403
                                0404
                                0406
                                0408
                                0410
                                0411
                                0413
                                0414
                                0416
```


0044	8F	00	6E		00	04 00076	RET			
					2C	00077	6\$:	MOVCS	#0, (SP), #0, #68, \$RMS_PTR	0429
					AE	0007E				
					8F	B0 00080		MOVW	#17409, \$RMS_PTR	
					8F	D0 00086		MOVL	#917504, \$RMS_PTR+4	
					01	90 0008E		MOVW	#1, \$RMS_PTR+30	
					8F	9B 00092		MOVZBW	#64, \$RMS_PTR+32	
					AE	9E 00097		MOVAB	REC_BUFFER, \$RMS_PTR+36	
					CF	9E 0009C		MOVAB	P.AAA, \$RMS_PTR+48	
					04	90 000A2		MOVW	#4, \$RMS_PTR+52	
					5E	DD 000A6		PUSHL	SP	0430
					AE	9F 000A8		PUSHAB	RAB+2	
					01	DD 000AB		PUSHL	#1	
					7E	D4 000AD		CLRL	-(SP)	
					04	FB 000AF		CALLS	#4, @#EXESOPEN_RDB	
					50	D0 000B6		MOVL	R0, STATUS	
					56	E8 000B9		BLBS	STATUS, 7\$	0431
					012C	31 000BC		BRW	21\$	
					AE	9F 000BF	7\$:	PUSHAB	RAB	0439
					01	FB 000C2		CALLS	#1, SYSSGET	
					50	D0 000C9		MOVL	R0, STATUS	
					56	E8 000CC		BLBS	STATUS, 9\$	0440
					00FE	31 000CF	8\$:	BRW	19\$	
					06	28 000D2	9\$:	MOVCS	#6, RAB+16, MAINT_RFA	0446
					0E	8A 000D8		BICB2	#14, RAB+6	0453
					8F	C8 000DC		BISL2	#1048584, RAB+6	0454
					8F	B0 000E4		MOVW	#544, RAB+52	0457
					AE	9E 000EA		MOVAB	NAME_BUFFER, RAB+48	0458
					AE	9F 000EF		PUSHAB	RAB	0459
					01	FB 000F2		CALLS	#1, SYSSFIND	
					50	D0 000F5		MOVL	R0, STATUS	
					56	E9 000F8		BLBC	STATUS, 10\$	0460
					8F	9A 000FB		MOVZBL	#148, STATUS	
					56	D1 000FF	10\$:	CMPL	STATUS, #98994	0461
					C7	12 00106		BNEQ	8\$	
					04	B0 00108		MOVW	#4, RAB+52	0472
					57	D5 0010C		TSTL	R7	0474
					25	13 0010E		BEQL	12\$	
					AE	9E 00110		MOVAB	LOC_ID, RAB+48	0477
					AE	9F 00115		PUSHAB	RAB	0478
					01	FB 00118		CALLS	#1, SYSSFIND	
					50	D0 0011B		MOVL	R0, STATUS	
					56	E9 0011E		BLBC	STATUS, 11\$	0479
					8F	3C 00121		MOVZWL	#8748, STATUS	
					56	D1 00126	11\$:	CMPL	STATUS, #98994	0480
					A0	12 0012D		BNEQ	8\$	
					57	D0 0012F		MOVL	R7, IDENTIFIER	0486
					64	11 00133		BRB	17\$	0474
					AE	D0 00135	12\$:	MOVL	REC_BUFFER+60, IDENTIFIER	0494
					AE	9E 0013A		MOVAB	IDENTIFIER, RAB+48	0501
					AE	D1 0013F	13\$:	CMPL	IDENTIFIER, #-1879048193	0504
					08	1B 00147		BLEQU	14\$	
					8F	D0 00149		MOVL	#-2147418112, IDENTIFIER	0506
					AE	9F 00151	14\$:	PUSHAB	RAB	0507
					01	FB 00154		CALLS	#1, SYSSFIND	
					50	D0 00157		MOVL	R0, STATUS	
					56	E8 0015A		BLBS	STATUS, 15\$	0508

PC	Op	Op2	Op3	Op4	Op5	Op6	Op7	Op8	Op9	Op10	Op11	Op12	Op13	Op14	Op15	Op16	Op17	Op18	Op19	Op20	Op21	Op22	Op23	Op24	Op25	Op26	Op27	Op28	Op29	Op30	Op31	Op32	Op33	Op34	Op35	Op36	Op37	Op38	Op39	Op40	Op41	Op42	Op43	Op44	Op45	Op46	Op47	Op48	Op49	Op50	Op51	Op52	Op53	Op54	Op55	Op56	Op57	Op58	Op59	Op60	Op61	Op62	Op63	Op64	Op65	Op66	Op67	Op68	Op69	Op70	Op71	Op72	Op73	Op74	Op75	Op76	Op77	Op78	Op79	Op80	Op81	Op82	Op83	Op84	Op85	Op86	Op87	Op88	Op89	Op90	Op91	Op92	Op93	Op94	Op95	Op96	Op97	Op98	Op99	Op100	Op101	Op102	Op103	Op104	Op105	Op106	Op107	Op108	Op109	Op110	Op111	Op112	Op113	Op114	Op115	Op116	Op117	Op118	Op119	Op120	Op121	Op122	Op123	Op124	Op125	Op126	Op127	Op128	Op129	Op130	Op131	Op132	Op133	Op134	Op135	Op136	Op137	Op138	Op139	Op140	Op141	Op142	Op143	Op144	Op145	Op146	Op147	Op148	Op149	Op150	Op151	Op152	Op153	Op154	Op155	Op156	Op157	Op158	Op159	Op160	Op161	Op162	Op163	Op164	Op165	Op166	Op167	Op168	Op169	Op170	Op171	Op172	Op173	Op174	Op175	Op176	Op177	Op178	Op179	Op180	Op181	Op182	Op183	Op184	Op185	Op186	Op187	Op188	Op189	Op190	Op191	Op192	Op193	Op194	Op195	Op196	Op197	Op198	Op199	Op200	Op201	Op202	Op203	Op204	Op205	Op206	Op207	Op208	Op209	Op210	Op211	Op212	Op213	Op214	Op215	Op216	Op217	Op218	Op219	Op220	Op221	Op222	Op223	Op224	Op225	Op226	Op227	Op228	Op229	Op230	Op231	Op232	Op233	Op234	Op235	Op236	Op237	Op238	Op239	Op240	Op241	Op242	Op243	Op244	Op245	Op246	Op247	Op248	Op249	Op250	Op251	Op252	Op253	Op254	Op255	Op256	Op257	Op258	Op259	Op260	Op261	Op262	Op263	Op264	Op265	Op266	Op267	Op268	Op269	Op270	Op271	Op272	Op273	Op274	Op275	Op276	Op277	Op278	Op279	Op280	Op281	Op282	Op283	Op284	Op285	Op286	Op287	Op288	Op289	Op290	Op291	Op292	Op293	Op294	Op295	Op296	Op297	Op298	Op299	Op300	Op301	Op302	Op303	Op304	Op305	Op306	Op307	Op308	Op309	Op310	Op311	Op312	Op313	Op314	Op315	Op316	Op317	Op318	Op319	Op320	Op321	Op322	Op323	Op324	Op325	Op326	Op327	Op328	Op329	Op330	Op331	Op332	Op333	Op334	Op335	Op336	Op337	Op338	Op339	Op340	Op341	Op342	Op343	Op344	Op345	Op346	Op347	Op348	Op349	Op350	Op351	Op352	Op353	Op354	Op355	Op356	Op357	Op358	Op359	Op360	Op361	Op362	Op363	Op364	Op365	Op366	Op367	Op368	Op369	Op370	Op371	Op372	Op373	Op374	Op375	Op376	Op377	Op378	Op379	Op380	Op381	Op382	Op383	Op384	Op385	Op386	Op387	Op388	Op389	Op390	Op391	Op392	Op393	Op394	Op395	Op396	Op397	Op398	Op399	Op400	Op401	Op402	Op403	Op404	Op405	Op406	Op407	Op408	Op409	Op410	Op411	Op412	Op413	Op414	Op415	Op416	Op417	Op418	Op419
----	----	-----	-----	-----	-----	-----	-----	-----	-----	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------

; Routine Size: 495 bytes, Routine Base: \$CODES\$ + 0191


```
578 0575 1 %SBTTL ' SYSS$CREATE_RDB - create rights data base'
579 0576 1 GLOBAL ROUTINE SYSS$CREATE_RDB (SYSID) =
580 0577 1
581 0578 1 ++
582 0579 1
583 0580 1 FUNCTIONAL DESCRIPTION:
584 0581 1
585 0582 1 This routine creates a new rights database. After creation the
586 0583 1 database contains the maintenance record and records for the
587 0584 1 environmental rights.
588 0585 1
589 0586 1 CALLING SEQUENCE:
590 0587 1 SYSS$CREATE_RDB (SYSID)
591 0588 1
592 0589 1 INPUT PARAMETERS:
593 0590 1 SYSID: (optional) address of the quadword system identifier
594 0591 1 to store in the maintenance record
595 0592 1
596 0593 1 IMPLICIT INPUTS:
597 0594 1 NONE
598 0595 1
599 0596 1 OUTPUT PARAMETERS:
600 0597 1 NONE
601 0598 1
602 0599 1 IMPLICIT OUTPUTS:
603 0600 1 NONE
604 0601 1
605 0602 1 ROUTINE VALUE:
606 0603 1 Status value of operation
607 0604 1
608 0605 1 SIDE EFFECTS:
609 0606 1 All active streams terminated, rights cache flushed,
610 0607 1 rights database created and opened
611 0608 1
612 0609 1 --
613 0610 1
614 0611 2 BEGIN
615 0612 2
616 0613 2 REGISTER
617 0614 2 SIZE = 1; ! size returned from EXE$ALOP1IMAG
618 0615 2 ADDRESS = 2; ! address returned from EXE$ALOP1IMAG
619 0616 2
620 0617 2 LOCAL
621 0618 2 LOC SYSID : REF VECTOR, ! local copy of SYSID
622 0619 2 STATUS : LONG, ! general status value
623 0620 2 CLOSE : BYTE, ! close rights database flag
624 0621 2 MAINT_RECORD : $BBLOCK [KGB$K MAINT_RECORD],
625 0622 2 ! buffer to build maintenance record
626 0623 2 FAB : $FAB_DECL, ! FAB to create rights database
627 0624 2 RAB : $RAB_DECL, ! RAB for rights database
628 0625 2 KEY0 : $XABKEY_DECL, ! XAB for primary key (identifier)
629 0626 2 KEY1 : $XABKEY_DECL, ! XAB for holder key
630 0627 2 KEY2 : $XABKEY_DECL, ! XAB for name key
631 0628 2 PROTECT : $XABPRO_DECL, ! XAB for file protection
632 0629 2 ARGLIST : VECTOR [2] ! argument list for EXE$SET_RDIPT
633 0630 2 INITIAL (1,0);
634 0631 2
```

```

635 0632 2 LABEL
636 0633 ~~~~~
637 0634 ~~~~~
638 0635 ~~~~~
639 0636 ~~~~~
640 0637 ~~~~~
641 0638 ~~~~~
642 0639 ~~~~~
643 0640 ~~~~~
644 0641 ~~~~~
645 0642 ~~~~~
646 0643 ~~~~~
647 0644 ~~~~~
648 0645 ~~~~~
649 P 0646 ~~~~~
650 P 0647 ~~~~~
651 P 0648 ~~~~~
652 P 0649 ~~~~~
653 0650 ~~~~~
654 0651 ~~~~~
655 0652 ~~~~~
656 0653 ~~~~~
657 0654 ~~~~~
658 0655 ~~~~~
659 0656 ~~~~~
660 0657 ~~~~~
661 0658 ~~~~~
662 0659 ~~~~~
663 0660 ~~~~~
664 0661 ~~~~~
665 0662 ~~~~~
666 0663 ~~~~~
667 0664 ~~~~~
668 0665 ~~~~~
669 0666 ~~~~~
670 0667 ~~~~~
671 0668 ~~~~~
672 0669 ~~~~~
673 0670 ~~~~~
674 0671 ~~~~~
675 0672 ~~~~~
676 0673 ~~~~~
677 0674 ~~~~~
678 0675 ~~~~~
679 0676 ~~~~~
680 0677 ~~~~~
681 0678 ~~~~~
682 0679 ~~~~~
683 P 0680 ~~~~~
684 P 0681 ~~~~~
685 P 0682 ~~~~~
686 P 0683 ~~~~~
687 P 0684 ~~~~~
688 P 0685 ~~~~~
689 P 0686 ~~~~~
690 P 0687 ~~~~~
691 P 0688 ~~~~~

! rights database is open in this block

! Validate parameters
!
LOC_SYSID = .SYSID;
IF .LOC_SYSID NEQU 0 AND NOT PROBER (%REF(0), %REF(8), .LOC_SYSID)
THEN
    RETURN SS$_ACCVIO;
! Do not open if file already exists
!
SFAB_INIT (FAB = FAB,
           FNM = 'RIGHTSLIST',
           DNM = 'SYSS$SYSTEM:.DAT',
           FAC = GET,
           SHR = (GET, PUT, DEL, UPD) );
STATUS = $OPEN(FAB=FAB);
IF .STATUS THEN
    RETURN RMS$_FEX;
! Allocate RDI if it has not been allocated already
!
IF .CTL$GL_RDIPTR EQLU 0
THEN
    BEGIN
        STATUS = EX$ALOP1IMAG (RDI$$RDIDEF, SIZE, ADDRESS);
        IF NOT .STATUS THEN RETURN SS$_INSFMEM;
        .ADDRESS = .SIZE;
        ARGLIST[1] = .ADDRESS;
        STATUS = SYSS$CMKRNL(EX$SET_RDIPTR, ARGLIST);
        IF NOT .STATUS THEN RETURN .STATUS;
        CH$FILL (0, RDI$$RDIDEF-4, .CTL$GL_RDIPTR+4);
    END
! Else Close out all active streams to the rights database
!
ELSE
    EX$CLOSE_RDB();
! Now set up the FAB and XAB's and create the file.
!
SFAB_INIT (FAB = FAB,
           FNM = 'RIGHTSLIST',
           DNM = 'SYSS$SYSTEM:.DAT',
           ORG = IDX,
           RFM = VAR,
           MRS = KGB$K_MAINT_RECORD,
           BKS = 2048,
           XAB = KEY0,
           FOP = (CBT, DFW),
```



```

692 P 0689 2 FAC = (GET, PUT, DEL, UPD),
693 P 0690 2 SHR = (GET, PUT, DEL, UPD),
694 0691 2 );
695 0692 2 FAB[FAB$V_LNM_MODE] = PSL$C_EXEC;
696 0693 2
697 P 0694 2 $XABKEY_INIT (
698 P 0695 2 XAB = KEY0,
699 P 0696 2 KREF = 0,
700 P 0697 2 KNM = UPLIT BYTE ('IDENTIFIER'),
701 P 0698 2 POS = $BYTEOFFSET (KGB$L_IDENTIFIER),
702 P 0699 2 SIZ = 4,
703 P 0700 2 DTP = BN4,
704 P 0701 2 FLG = DUP,
705 P 0702 2 NXT = KEY1,
706 0703 2 );
707 0704 2
708 P 0705 2 $XABKEY_INIT (
709 P 0706 2 XAB = KEY1,
710 P 0707 2 KREF = 1,
711 P 0708 2 KNM = UPLIT BYTE ('HOLDER'),
712 P 0709 2 POS = $BYTEOFFSET (KGB$Q HOLDER),
713 P 0710 2 SIZ = 8,
714 P 0711 2 DTP = STG,
715 P 0712 2 FLG = (DUP, NUL, CHG),
716 P 0713 2 NUL = 0,
717 P 0714 2 NXT = KEY2,
718 0715 2 );
719 0716 2
720 P 0717 2 $XABKEY_INIT (
721 P 0718 2 XAB = KEY2,
722 P 0719 2 KREF = 2,
723 P 0720 2 KNM = UPLIT BYTE ('NAME'),
724 P 0721 2 POS = $BYTEOFFSET (KGB$T_NAME),
725 P 0722 2 SIZ = KGB$S_NAME,
726 P 0723 2 DTP = STG,
727 P 0724 2 FLG = (NUL, CHG),
728 P 0725 2 NUL = 0,
729 P 0726 2 NXT = PROTECT,
730 0727 2 );
731 0728 2
732 P 0729 2 $XABPRO_INIT (
733 P 0730 2 XAB = PROTECT,
734 P 0731 2 PRO = (RWED, RWED, R, R),
735 P 0732 2 UIC = (1,4),
736 0733 2 );
737 0734 2
738 0735 2 IF .CTL$GL_IMGHDRBF EQLU 0
739 0736 2 THEN
740 0737 2 BEGIN
741 0738 2 CLOSE = 1;
742 0739 2 FAB[FAB$V_PPF] = 1;
743 0740 2 END
744 0741 2 ELSE
745 0742 2 CLOSE = 0;
746 0743 2 STATUS = $CREATE (FAB = FAB);
747 0744 2 IF NOT .STATUS THEN RETURN .STATUS;
748 0745 2
```

```

: 746 0746 2 RDB_OPEN:
: 750 0747 BEGIN
: 751 0748 CTL$GL_RDIPT[RDISL_IFI_WRITE] = .FAB[FAB$W_IFI];
: 752 0749
: 753 0750 ! Now set up and connect a RAB, and write the maintenance record.
: 754 0751 !
: 755 0752
: 756 P 0753 $RAB_INIT (RAB = RAB,
: 757 P 0754 FAB = FAB,
: 758 P 0755 RAC = KEY,
: 759 P 0756 RBF = MAINT_RECORD,
: 760 P 0757 RSZ = KGB$K_MAINT_RECORD
: 761 0758 );
: 762 0759
: 763 0760 STATUS = $CONNECT (RAB = RAB);
: 764 0761 IF NOT .STATUS THEN LEAVE RDB_OPEN;
: 765 0762 VECTOR [CTL$GL_RDIPT[RDISL_ISI_VEC], 0] = .RAB[RAB$W_ISI];
: 766 0763
: 767 0764 CH$FILL (0, KGB$K_MAINT_RECORD, MAINT_RECORD);
: 768 0765 CH$MOVE (KGB$S_NAME, UPLIT BYTE ('$MAINTENANCE_RECORD '),
: 769 0766 MAINT_RECORD[KGB$T_NAME]);
: 770 0767 IF .LOC_SYSID NEQ 0
: 771 0768 THEN
: 772 0769 CH$MOVE (KGB$S_SYS_ID, .LOC_SYSID, MAINT_RECORD[KGB$Q_SYS_ID])
: 773 0770 ELSE
: 774 0771 $GETTIM (TIMADR = MAINT_RECORD[KGB$Q_SYS_ID]);
: 775 0772 MAINT_RECORD[KGB$W_LEVEL] = KGB$K_LEVEL1;
: 776 0773 MAINT_RECORD[KGB$L_NEXT_ID] = UIC$K_FIRST_ID;
: 777 0774
: 778 0775 STATUS = $PUT (RAB = RAB);
: 779 0776 IF NOT .STATUS THEN LEAVE RDB_OPEN;
: 780 0777
: 781 0778 ! Create records for the environmental rights
: 782 0779 !
: 783 0780
: 784 0781 RAB[RAB$W_RSZ] = KGB$K_IDENT_RECORD;
: 785 0782
: 786 0783 MAINT_RECORD[KGB$L_IDENTIFIER] = KGB$K_BATCH_ID;
: 787 0784 CH$MOVE (KGB$S_NAME, UPLIT BYTE ('BATCH '),
: 788 0785 MAINT_RECORD[KGB$T_NAME]);
: 789 0786 STATUS = $PUT (RAB = RAB);
: 790 0787 IF NOT .STATUS THEN LEAVE RDB_OPEN;
: 791 0788
: 792 0789 MAINT_RECORD[KGB$L_IDENTIFIER] = KGB$K_DIALUP_ID;
: 793 0790 CH$MOVE (KGB$S_NAME, UPLIT BYTE ('DIALUP '),
: 794 0791 MAINT_RECORD[KGB$T_NAME]);
: 795 0792 STATUS = $PUT (RAB = RAB);
: 796 0793 IF NOT .STATUS THEN LEAVE RDB_OPEN;
: 797 0794
: 798 0795 MAINT_RECORD[KGB$L_IDENTIFIER] = KGB$K_INTERACTIVE_ID;
: 799 0796 CH$MOVE (KGB$S_NAME, UPLIT BYTE ('INTERACTIVE '),
: 800 0797 MAINT_RECORD[KGB$T_NAME]);
: 801 0798 STATUS = $PUT (RAB = RAB);
: 802 0799 IF NOT .STATUS THEN LEAVE RDB_OPEN;
: 803 0800
: 804 0801 MAINT_RECORD[KGB$L_IDENTIFIER] = KGB$K_LOCAL_ID;
: 805 0802 CH$MOVE (KGB$S_NAME, UPLIT BYTE ('LOCAL '),
```



```

: 806      0803      3      MAINT_RECORD[KGB$T_NAME]);
: 807      0804      3      STATUS = $PUT (RAB = RAB);
: 808      0805      3      IF NOT .STATUS THEN LEAVE RDB_OPEN;
: 809      0806      3
: 810      0807      3      MAINT_RECORD[KGB$L_IDENTIFIER] = KGB$K_NETWORK_ID;
: 811      0808      3      CH$MOVE (KGB$S_NAME, UPLIT BYTE ('NETWORK
: 812      0809      3      MAINT_RECORD[KGB$T_NAME]));
: 813      0810      3      STATUS = $PUT (RAB = RAB);
: 814      0811      3      IF NOT .STATUS THEN LEAVE RDB_OPEN;
: 815      0812      3
: 816      0813      3      MAINT_RECORD[KGB$L_IDENTIFIER] = KGB$K_REMOTE_ID;
: 817      0814      3      CH$MOVE (KGB$S_NAME, UPLIT BYTE ('REMOTE
: 818      0815      3      MAINT_RECORD[KGB$T_NAME]));
: 819      0816      3      STATUS = $PUT (RAB = RAB);
: 820      0817      3      IF NOT .STATUS THEN LEAVE RDB_OPEN;
: 821      0818      3
: 822      0819      3      STATUS = SSS_NORMAL;
: 823      0820      3      END;
: 824      0821      3
: 825      0822      3      IF .CLOSE THEN EXE$CLOSE_RDB();
: 826      0823      2      RETURN .STATUS;
: 827      0824      1      END;

```

! End of routine SYSS\$CREATE_RDB

```

.PSECT $SPLIT$,NOWRT,NOEXE,2
54 41 44 2E 3A 54 53 49 4C 53 54 48 47 49 52 00004 P.AAB: .ASCII \RIGHTSLIST\
20 20 20 20 20 4D 45 54 53 59 53 24 53 59 53 0000E P.AAC: .ASCII \SYSS$SYSTEM:.DAT\
20 20 20 20 20 54 53 49 4C 53 54 48 47 49 52 0001D P.AAD: .ASCII \RIGHTSLIST\
20 20 20 20 20 4D 45 54 53 59 53 24 53 59 53 00027 P.AAE: .ASCII \SYSS$SYSTEM:.DAT\
20 20 20 20 20 52 45 49 46 49 54 4E 45 44 49 00036 P.AAF: .ASCII \IDENTIFIER
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 00045
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 00054
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 00056 P.AAG: .ASCII \HOLDER
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 00065
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 00074
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 00076 P.AAH: .ASCII \NAME
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 00085
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 00094
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 00096 P.AAI: .ASCII \$$MAINTENANCE_RECORD
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 000A5
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 000B4
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 000B6 P.AAJ: .ASCII \BATCH
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 000C5
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 000D4
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 000D6 P.AAK: .ASCII \DIALUP
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 000E5
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 000F4
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 000F6 P.AAL: .ASCII \INTERACTIVE
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 00105
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 00114
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 00116 P.AAM: .ASCII \LOCAL
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 00125
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 00134
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 00136 P.AAN: .ASCII \NETWORK
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 00145

```

20	20	20	20	20	20	20	20	20	20	45	54	4F	4D	20	20	00154
20	20	20	20	20	20	20	20	20	20	20	20	20	20	45	52	00156
														20	20	00165
														20	20	00174

P.AAO: .ASCII \REMOTE

.EXTRN SYSS\$OPEN, SYSS\$CREATE
.EXTRN SYSS\$CONNECT, SYSS\$GETTIM

.PSECT \$CODE\$,NOWRT,2

.ENTRY SYSS\$CREATE_RDB, Save R2,R3,R4,R5,R6,R7,R8,- : 0576
R9,R10,R11

MOVAB P.AAB, R11

MOVAB SYSS\$PUT, R10

MOVAB -532(SP), SP

PUSHL #1

CLRL ARGLIST+4

MOVL SYSID, LOC_SYSID

CLRL R9

TSTL LOC_SYSID

BEQL 1\$

INCL R9

PROBER #0, #8, (LOC_SYSID)

BNEQ 1\$

MOVL #12, R0

RET

MOVCS #0, (SP), #0, #80, \$RMS_PTR

MOVW #20483, \$RMS_PTR

MOVW #3842, \$RMS_PTR+22

MOVB #2, \$RMS_PTR+31

MOVAB P.AAB, \$RMS_PTR+44

MOVAB P.AAC, \$RMS_PTR+48

MOVW #3850, \$RMS_PTR+52

PUSHAB FAB

CALLS #1, SYSS\$OPEN

MOVL R0, STATUS

BLBC STATUS, 2\$

MOVL #98946, R0

RET

TSTL @#CTL\$GL_RDIPT

BNEQ 5\$

MOVL #56, R1

JSB @#EXE\$ALOP1IMAG

MOVL R0, STATUS

BLBS STATUS, 3\$

MOVZWL #292, R0

RET

MOVL SIZE, (ADDRESS)

MOVL ADDRESS, ARGLIST+4

PUSHL SP

PUSHL #EXE\$SET_RDIPT

CALLS #2, @#SYSS\$CMKRN

MOVL R0, STATUS

BLBS STATUS, 4\$

BRW 16\$

MOVL @#CTL\$GL_RDIPT, R0

OFFC 00000

5B 0000' CF 9E 00002

5A 00000000G 00 9E 00007

5E FDEC CE 9E 0000E

01 DD 00013

04 AE D4 00015

58 04 AC D0 00018

59 D4 0001C

58 D5 0001E

0C 13 00020

59 D6 00022

68 08 00 0C 00024

04 12 00028

50 0C D0 0002A

04 0002D

0050 8F 00 6E 2C 0002E 1\$:

CD 00035

FF70 CD 5003 8F B0 00038

86 AD 0F02 8F B0 0003F

8F AD 02 90 00045

9C AD 6B 9E 00049

A0 AD 0A AB 9E 0004D

A4 AD 0F0A 8F B0 00052

FF70 CD 9F 00058

00000000G 00 01 FB 0005C

56 50 D0 00063

08 56 E9 00066

50 00018282 8F D0 00069

04 00070

00000000G 9F D5 00071 2\$:

44 12 00077

51 38 D0 00079

00000000G 9F 16 0007C

56 50 D0 00082

06 56 E8 00085

50 0124 8F 3C 00088

04 0008D

04 62 51 D0 0008E 3\$:

AE 52 D0 00091

5E DD 00095

00000000G 8F DD 00097

00000000G 9F 02 FB 0009D

56 50 D0 000A4

03 56 E8 000A7

0278 31 000AA

50 00000000G 9F D0 000AD 4\$:

	34	00	6E		04	00	2C	000B4	MOVCS	#0, (SP), #0, #52, 4(R0)	
						A0		000B9			
						07	11	000BB	BRB	6\$	0659
0050	8F	00	00000000G	9F		00	FB	000BD	CALLS	#0, @#EXESCLOSE RDB	0675
				6E		00	2C	000C4	MOVCS	#0, (SP), #0, #80, \$RMS_PTR	0691
					FF70	CD		000CB			
					5003	8F	B0	000CE	MOVW	#20483, \$RMS_PTR	
					00200020	8F	D0	000D5	MOVL	#2097184, \$RMS_PTR+4	
					0F0F	8F	B0	000DE	MOVW	#3855, \$RMS_PTR+22	
						20	90	000E4	MOVB	#32, \$RMS_PTR+29	
						02	90	000E8	MOVB	#2, \$RMS_PTR+31	
					00F8	CE	9E	000EC	MOVAB	KEY0, \$RMS_PTR+36	
					19	AB	9E	000F2	MOVAB	P.AAD, \$RMS_PTR+44	
					23	AB	9E	000F7	MOVAB	P.AAE, \$RMS_PTR+48	
					00400FOA	8F	D0	000FC	MOVL	#4198154, \$RMS_PTR+52	
					AE	AD	94	00104	CLRB	\$RMS_PTR+62	
BA	AD	02		00		01	F0	00107	INSV	#1, #0, #2, FAB+74	0692
004C	8F	00		6E		00	2C	0010D	MOVCS	#0, (SP), #0, #76, \$RMS_PTR	0703
					00F8	CE		00114			
					4C15	8F	B0	00117	MOVW	#19477, \$RMS_PTR	
					00AC	CE	9E	0011E	MOVAB	KEY1, \$RMS_PTR+4	
					0401	8F	B0	00125	MOVW	#1025, \$RMS_PTR+18	
						04	90	0012C	MOVB	#4, \$RMS_PTR+46	
004C	8F	00		32		AB	9E	00131	MOVAB	P.AAF, \$RMS_PTR+56	
				6E		00	2C	00137	MOVCS	#0, (SP), #0, #76, \$RMS_PTR	0715
					00AC	CE		0013E			
					4C15	8F	B0	00141	MOVW	#19477, \$RMS_PTR	
					60	AE	9E	00148	MOVAB	KEY2, \$RMS_PTR+4	
						07	B0	0014E	MOVW	#7, \$RMS_PTR+18	
						01	90	00153	MOVB	#1, \$RMS_PTR+23	
						08	B0	00158	MOVW	#8, \$RMS_PTR+30	
						08	90	0015D	MOVB	#8, \$RMS_PTR+46	
004C	8F	00		52		AB	9E	00162	MOVAB	P.AAG, \$RMS_PTR+56	
				6E		00	2C	00168	MOVCS	#0, (SP), #0, #76, \$RMS_PTR	0727
					60	AE		0016F			
					4C15	8F	B0	00171	MOVW	#19477, \$RMS_PTR	
					08	AE	9E	00177	MOVAB	PROTECT, \$RMS_PTR+4	
						06	B0	0017C	MOVW	#6, \$RMS_PTR+18	
						02	90	00180	MOVB	#2, \$RMS_PTR+23	
						10	B0	00184	MOVW	#16, \$RMS_PTR+30	
						20	90	00188	MOVB	#32, \$RMS_PTR+46	
0058	8F	00		72		AB	9E	0018D	MOVAB	P.AAH, \$RMS_PTR+56	
				6E		00	2C	00193	MOVCS	#0, (SP), #0, #88, \$RMS_PTR	0733
					08	AE		0019A			
					5813	8F	B0	0019C	MOVW	#22547, \$RMS_PTR	
					EE00	8F	B0	001A2	MOVW	#-4608, \$RMS_PTR+8	
					00010004	8F	D0	001A8	MOVL	#65540, \$RMS_PTR+12	
					00000000G	9F	D5	001B0	TSTL	@#CTL\$GL_IMGDRBF	0735
						0A	12	001B6	BNEQ	7\$	

0044	8F	00	08	50	00000000G	014D	31	001D5	BRW	16\$			
				A0	FF72	9F	D0	001D8	9\$:	MOVZWL	#CTL\$GL RDIPTR, R0	0748	
				6E		CD	3C	001DF		MOVZWL	FAB+2, 8(R0)		
					FF2C	00	2C	001E5		MOVZWL	#0, (SP), #0, #68, \$RMS_PTR	0758	
					4401	CD		001EC					
				FF2C		8F	B0	001EF		MOVW	#17409, \$RMS_PTR		
				FF4A		01	90	001F6		MOVW	#1, \$RMS_PTR+30		
				FF4E		8F	9B	001FB		MOVZBW	#64, \$RMS_PTR+34		
				FF54		AD	9E	00201		MOVAB	MAINT_RECORD, \$RMS_PTR+40		
				FF68		CD	9E	00207		MOVAB	FAB, \$RMS_PTR+60		
					FF2C	CD	9F	0020E		PUSHAB	RAB	0760	
				00000000G		01	FB	00212		CALLS	#1, SYSS\$CONNECT		
						50	D0	00219		MOVW	R0, STATUS		
						56	E9	0021C		BLBC	STATUS, 12\$	0761	
						9F	D0	0021F		MOVW	#CTL\$GL RDIPTR, R0	0762	
				0C	FF2E	CD	3C	00226		MOVZWL	RAB+2, 12(R0)		
0040	8F	00		6E		00	2C	0022C		MOVZWL	#0, (SP), #0, #64, MAINT_RECORD	0764	
					C0	AD		00233					
				DO	AD	0092	CB	20	28	00235	MOVZC3	#32, P.AAI, MAINT_RECORD+16	0766
						07		59	E9	0023C	BLBC	R9, 10\$	0767
				F4	AD		68	08	28	0023F	MOVZC3	#8, (LOC_SYSID), MAINT_RECORD+52	0769
								0A	11	00244	BRB	11\$	
					F4	AD	9F	00246	10\$:	PUSHAB	MAINT_RECORD+52	0771	
					00000000G	00	01	FB	00249	CALLS	#1, SYSS\$GETTIM		
					F0	AD	8F	B0	00250	MOVW	#257, MAINT_RECORD+48	0772	
					FC	AD	8F	D0	00256	MOVW	#-2147418112, MAINT_RECORD+60	0773	
					FF2C	CD	9F	0025E		PUSHAB	RAB	0775	
						01	FB	00262		CALLS	#1, SYSS\$PUT		
						50	D0	00265		MOVW	R0, STATUS		
						56	E9	00268		BLBC	STATUS, 13\$	0776	
						30	B0	0026B		MOVW	#48, RAB+34	0781	
					FF4E	CD	8F	D0	00270	MOVW	#-2147483647, MAINT_RECORD	0783	
					C0	AD	20	28	00278	MOVZC3	#32, P.AAJ, MAINT_RECORD+16	0785	
DO	AD	00B2		CB		CD	9F	0027F		PUSHAB	RAB	0786	
					FF2C	01	FB	00283		CALLS	#1, SYSS\$PUT		
						50	D0	00286		MOVW	R0, STATUS		
						56	E9	00289	12\$:	BLBC	STATUS, 14\$	0787	
						8F	D0	0028C		MOVW	#-2147483646, MAINT_RECORD	0789	
DO	AD	00D2		CB		20	28	00294		MOVZC3	#32, P.AAK, MAINT_RECORD+16	0791	
					FF2C	CD	9F	0029B		PUSHAB	RAB	0792	
						01	FB	0029F		CALLS	#1, SYSS\$PUT		
						50	D0	002A2		MOVW	R0, STATUS		
						56	E9	002A5		BLBC	STATUS, 15\$	0793	
						8F	D0	002A8		MOVW	#-2147483645, MAINT_RECORD	0795	
DO	AD	00F2		CB		20	28	002B0		MOVZC3	#32, P.AAL, MAINT_RECORD+16	0797	
					FF2C	CD	9F	002B7		PUSHAB	RAB	0798	
						01	FB	002BB		CALLS	#1, SYSS\$PUT		
						50	D0	002BE		MOVW	R0, STATUS		
						56	E9	002C1		BLBC	STATUS, 15\$	0799	
						8F	D0	002C4		MOVW	#-2147483644, MAINT_RECORD	0801	
DO	AD	0112		CB		20	28	002CC		MOVZC3	#32, P.AAM, MAINT_RECORD+16	0803	
					FF2C	CD	9F	002D3		PUSHAB	RAB	0804	
						01	FB	002D7		CALLS	#1, SYSS\$PUT		
						50	D0	002DA		MOVW	R0, STATUS		
						56	E9	002DD	13\$:	BLBC	STATUS, 15\$	0805	
						8F	D0	002E0		MOVW	#-2147483643, MAINT_RECORD	0807	
DO	AD	0132		CB		20	28	002E8		MOVZC3	#32, P.AAN, MAINT_RECORD+16	0809	

RDBSHR
V04-000

RDBSHR - Rights database loadable system service
SYSS\$CREATE_RDB - create rights data base

H 9
16-Sep-1984 01:48:50
14-Sep-1984 12:40:52

VAX-11 Bliss-32 V4.0-742
[LOADSS.SRC]RDBSHR.B32;1

Page 25
(4)

			6A	FF2C	CD	9F	002EF		PUSHAB	RAB		0810
			56		01	FB	002F3		CALLS	#1, SYSS\$PUT		
			1F		50	D0	002F6		MOVL	R0, STATUS		
			AD	80000006	56	E9	002F9	14\$:	BLBC	STATUS, 15\$		0811
DO	AD	C0	CB		8F	D0	002FC		MOVL	#-2147483642, MAINT_RECORD		0813
		0152			20	28	00304		MOVC3	#32, P.AAO, MAINT_RECORD+16		0815
				FF2C	CD	9F	0030B		PUSHAB	RAB		0816
			6A		01	FB	0030F		CALLS	#1, SYSS\$PUT		
			56		50	D0	00312		MOVL	R0, STATUS		
			03		56	E9	00315		BLBC	STATUS, 15\$		0817
			56		01	D0	00318		MOVL	#1, STATUS		0819
			07		57	E9	0031B	15\$:	BLBC	CLOSE, 16\$		0822
		00000000G	9F		00	FB	0031E		CALLS	#0, @EXE\$CLOSE_RDB		
			50		56	D0	00325	16\$:	MOVL	STATUS, R0		0823
					04	00328			RET			0824

; Routine Size: 809 bytes, Routine Base: \$CODE\$ + 0380

```
0825 1 %SBTTL ' SYSS$FIND_HOLDER - search RDB for ident holders'
0826 1 GLOBAL ROUTINE SYSS$FIND_HOLDER (ID, HOLDER, ATTRIB, CONTXT) =
0827 1
0828 1 ++
0829 1
0830 1 FUNCTIONAL DESCRIPTION:
0831 1
0832 1 This routine searches the rights database for all holders
0833 1 of the specified identifier, and returns their identifier and
0834 1 attributes.
0835 1
0836 1 CALLING SEQUENCE:
0837 1 SYSS$FIND_HOLDER (ID, HOLDER, ATTRIB, CONTXT)
0838 1
0839 1 INPUT PARAMETERS:
0840 1 ID: identifier longword whose holder records
0841 1 are to be found
0842 1 CONTXT: (optional) address of a longword containing the
0843 1 record stream context. initially should be zero,
0844 1 value output on first call, value input on
0845 1 subsequent calls.
0846 1
0847 1 IMPLICIT INPUTS:
0848 1 NONE
0849 1
0850 1 OUTPUT PARAMETERS:
0851 1 HOLDER: (optional) address to return the holder id quadword
0852 1 ATTRIB: (optional) address to return the attributes longword
0853 1
0854 1 IMPLICIT OUTPUTS:
0855 1 NONE
0856 1
0857 1 ROUTINE VALUE:
0858 1 Status of operation
0859 1
0860 1 SIDE EFFECTS:
0861 1 NONE
0862 1
0863 1 --
0864 1
0865 2 BEGIN
0866 2
0867 2 LOCAL
0868 2 LOC_ID : LONG, ! local copy of ID
0869 2 LOC_HOLDER : LONG, ! local copy of HOLDER
0870 2 LOC_ATTRIB : LONG, ! local copy of ATTRIB
0871 2 LOC_CONTXT : LONG, ! local copy of CONTXT
0872 2 STATUS : LONG, ! general status value
0873 2 CONTINUE : LONG, ! flag indicating continuation
0874 2 CLOSE : LONG, ! call to EXE$CLOSE_RDB required flag
0875 2 RAB : $RAB DECL, ! RAB for file I/O
0876 2 REC_BUFFER : $BBLOCK [KGB$K_IDENT_RECORD]; ! record buffer to read records
0877 2
0878 2
0879 2
0880 2 LABEL
0881 2 RDB_OPEN; ! rights database is open in this block
```



```

886 0882 2
887 0883 2 ! Validate parameters
888 0884 2 !
889 0885 2
890 0886 2 LOC_ID = .ID;
891 0887 2 IF (.LOC_ID AND UIC$M_ID_FORM_FLAG) NEQU 0
892 0888 2 THEN
893 0889 2     (IF (.LOC_ID GTRU UIC$K_LAST_ID) THEN RETURN SS$_IVIDENT)
894 0890 2 ELSE
895 0891 2     (IF (.LOC_ID GTRU UIC$K_MAX_UIC) OR (.LOC_ID EQL 0) THEN RETURN SS$_IVIDENT);
896 0892 2
897 0893 2 LOC_HOLDER = .HOLDER;
898 0894 2 IF .LOC_HOLDER NEQU 0 AND NOT PROBEW (%REF(0), %REF(8), .LOC_HOLDER)
899 0895 2 THEN
900 0896 2     RETURN SS$_ACCVIO;
901 0897 2
902 0898 2 LOC_ATTRIB = .ATTRIB;
903 0899 2 IF .LOC_ATTRIB NEQU 0 AND NOT PROBEW (%REF(0), %REF(4), .LOC_ATTRIB)
904 0900 2 THEN
905 0901 2     RETURN SS$_ACCVIO;
906 0902 2
907 0903 2 LOC_CONTXT = .CONTXT;
908 0904 2 IF .LOC_CONTXT NEQU 0 AND NOT PROBEW (%REF(0), %REF(4), .LOC_CONTXT)
909 0905 2 THEN
910 0906 2     RETURN SS$_ACCVIO;
911 0907 2
912 0908 2 ! Open the rights database for reading. Record whether this is an initial
913 0909 2 ! call or a continuation by checking if the context is zero or not.
914 0910 2 !
915 0911 2
916 0912 2 CONTINUE = (IF .LOC_CONTXT NEQU 0 THEN ..LOC_CONTXT NEQU 0 ELSE 0);
917 0913 2
918 P 0914 2 $RAB_INIT (RAB = RAB,
919 P 0915 2     RAC = KEY,
920 P 0916 2     KRF = 0,
921 P 0917 2     KSZ = 4,
922 P 0918 2     KBF = LOC_ID,
923 P 0919 2     ROP = (WAT, NLK, LIM),
924 P 0920 2     USZ = KGB$K_IDENT_RECORD,
925 P 0921 2     UBF = REC_BUFFER
926 0922 2 );
927 0923 2 STATUS = EXE$OPEN RDB (.LOC_CONTXT, 0, RAB[RAB$W_ISI], CLOSE);
928 0924 2 IF NOT .STATUS THEN RETURN .STATUS;
929 0925 2
930 0926 2 RDB_OPEN:
931 0927 2     BEGIN
932 0928 2
933 0929 2     ! On an initial call, do an indexed $GET to position to the identifier
934 0930 2     ! record.
935 0931 2     !
936 0932 2
937 0933 2     IF NOT .CONTINUE
938 0934 2     THEN
939 0935 2         BEGIN
940 0936 2         STATUS = $GET (RAB = RAB);
941 0937 2         IF .STATUS EQLU RMSS$_RNF THEN STATUS = SS$_NOSUCHID;
942 0938 2         IF NOT .STATUS
```

```

: 943 0939 4      THEN
: 944 0940 3      BEGIN
: 945 0941 3      EXE$$FINISH_RDB (.LOC_CONXT);
: 946 0942 3      LEAVE RDB_OPEN;
: 947 0943 4      END;
: 948 0944 3      END;
: 949 0945 3
: 950 0946 3      ! Switch to sequential mode and read the next holder record, and
: 951 0947 3      ! return the data items.
: 952 0948 3
: 953 0949 3
: 954 0950 3      RAB[RAB$B_RAC] = RAB$C_SEQ;
: 955 0951 3      STATUS = $GET (RAB = RAB);
: 956 0952 3      IF .STATUS EQLU RMSS_EOF OR .STATUS EQLU RMSS_OK_LIM
: 957 0953 3      THEN
: 958 0954 3          STATUS = SS$_NOSUCHID;
: 959 0955 3      IF NOT .STATUS
: 960 0956 3      THEN
: 961 0957 4          BEGIN
: 962 0958 4          EXE$$FINISH_RDB (.LOC_CONXT);
: 963 0959 4          LEAVE RDB_OPEN;
: 964 0960 4          END;
: 965 0961 3
: 966 0962 3      IF .LOC_HOLDER NEQU 0
: 967 0963 3      THEN
: 968 0964 3          CH$MOVE (KGB$$_HOLDER, REC_BUFFER[KGB$Q_HOLDER], .LOC_HOLDER);
: 969 0965 3      IF .LOC_ATTRIB NEQU 0
: 970 0966 3      THEN
: 971 0967 3          .LOC_ATTRIB = .REC_BUFFER[KGB$L_ATTRIBUTES];
: 972 0968 3
: 973 0969 3      STATUS = SS$_NORMAL;
: 974 0970 3      END;
: 975 0971 3
: 976 0972 3      ! Close the rights database if there is no image
: 977 0973 3      !
: 978 0974 3
: 979 0975 2      IF .CLOSE THEN EXE$CLOSE_RDB();
: 980 0976 2      RETURN .STATUS
: 981 0977 1      END;
                                ! End of routine SYSS$FIND_HOLDER
```

			OFFC	00000		.ENTRY	SYSS\$FIND_HOLDER, Save R2,R3,R4,R5,R6,R7,R8,-;	0826
						R9,R10,RT1		
	04	5E	80	AE	9E	00002	-128(SP), SP	
		AE	04	AC	D0	00006	ID, LOC_ID	0886
				0C	18	0000B	1\$	0887
8FFFFFFF	8F		04	AE	D1	0000D	LOC_ID, #-1879048193	0889
				17	1B	00015	3\$	
				0F	11	00017	2\$	
3FFFFFFF	8F		04	AE	D1	00019	LOC_ID, #1073741823	0891
				05	1A	00021	2\$	
			04	AE	D5	00023	LOC_ID	
				06	12	00026	3\$	
	50	2224	8F	3C	00028	2\$	#8740, R0	

				5A	08	AC	04	0002D	3\$:	RET	HOLDER, LOC_HOLDER	0893
						6E	D4	0002E		MOVL	(SP)	0894
						5A	D5	00032		CLRL	LOC_HOLDER	
						08	13	00034		TSTL	4\$	
						6E	D6	00036		BEQL	(SP)	
6A				08		00	0D	00038		INCL	#0, #8, (LOC_HOLDER)	
						24	13	0003A		PROBEW	6\$	
				59	0C	AC	D0	0003E	4\$:	BEQL	ATTRIB, LOC_ATTRIB	0898
						5B	D4	00040		MOVL	R11	0899
						59	D5	00044		CLRL	LOC_ATTRIB	
						08	13	00046		TSTL	5\$	
						5B	D6	00048		BEQL	R11	
69				04		00	0D	0004A		INCL	#0, #4, (LOC_ATTRIB)	
						12	13	0004C		PROBEW	6\$	
				57	10	AC	D0	00050	5\$:	BEQL	CONXT, LOC_CONXT	0903
						50	D4	00052		MOVL	R0	0904
						57	D5	00056		CLRL	LOC_CONXT	
						0C	13	00058		TSTL	7\$	
						50	D6	0005A		BEQL	R0	
67				04		00	0D	0005C		INCL	#0, #4, (LOC_CONXT)	
						04	12	0005E		PROBEW	7\$	
				50		0C	D0	00062	6\$:	BNEQ	#12, R0	0906
							04	00064		MOVL	R0, 9\$	0912
				0D		50	E9	00067	7\$:	RET	(LOC_CONXT)	
						50	D4	00068		BLBC	8\$	
						67	D5	0006B		CLRL	R0	
						02	13	0006D		TSTL	R0, CONTINUE	
						50	D6	0006F		BEQL	10\$	
				58		50	D0	00071	8\$:	INCL	CONTINUE	
						02	11	00073		MOVW	#0, (SP), #0, #68, \$RMS_PTR	0922
						58	D4	00076	9\$:	BRB		
						00	2C	00078	10\$:	CLRL		
0044	8F	00		6E		AE	00	0007A		MOVC5		
					3C	AE	00	00081				
					4401	8F	B0	00083		MOVW	#17409, \$RMS_PTR	
					00124000	8F	D0	00089		MOVL	#1196032, \$RMS_PTR+4	
						01	90	00091		MOVW	#1, \$RMS_PTR+30	
						30	B0	00095		MOVW	#48, \$RMS_PTR+32	
						AE	9E	00099		MOVAB	REC_BUFFER, \$RMS_PTR+36	
					0C	AE	9E	0009E		MOVAB	LOC_ID, \$RMS_PTR+48	
					04	AE	90	000A3		MOVW	#4, \$RMS_PTR+52	
						AE	9F	000A7		PUSHAB	CLOSE	0923
					08	AE	9F	000AA		PUSHAB	RAB+2	
					42	AE	9F	000AD		PUSHAB	- (SP)	
						7E	D4	000AF		CLRL	LOC_CONXT	
						57	DD	000B1		PUSHL	#4, @#EXESOPEN_RDB	
						04	FB	000B8		CALLS	R0, STATUS	
						50	D0	000BB		MOVL	STATUS, 20\$	0924
						56	E9	000BE		BLBC	CONTINUE, 12\$	0933
						58	E8	000C1		BLBS	RAB	0936
					3C	AE	9F	000C4		PUSHAB	#1, SYSSGET	
		</										

Address	Hex	Label	Instruction	Comment
00000000G	00	3C	AE 9F 000E2	PUSHAB RAB
	56		01 FB 000E5	CALLS #1, SYSSGET
0001827A	8F		50 D0 000EC	MOVL R0, STATUS
			56 D1 000EF	CMPL STATUS, #98938
00018051	8F		09 13 000F6	BEQL 13\$
			56 D1 000F8	CMPL STATUS, #98385
			05 12 000FF	BNEQ 14\$
	56	21EC	8F 3C 00101	MOVZWL #8684, STATUS
	0B		56 E8 00106	BLBS STATUS, 16\$
			57 DD 00109	15\$: PUSHL LOC_CONXT
00000000G	9F		01 FB 0010B	CALLS #1, @#EXE\$\$\$FINISH_RDB
			12 11 00112	BRB 19\$
	05		6E E9 00114	16\$: BLBC (SP), 17\$
6A 14	AE		08 28 00117	MOVC3 #8, REC_BUFFER+8, (LOC_HOLDER)
	04		5B E9 0011C	17\$: BLBC R11, 18\$
	69	10	AE D0 0011F	MOVL REC_BUFFER+4, (LOC_ATTRIB)
	56		01 D0 00123	18\$: MOVL #1, STATUS
	07	08	AE E9 00126	19\$: BLBC CLOSE, 20\$
00000000G	9F		00 FB 0012A	CALLS #0, @#EXE\$CLOSE_RDB
	50		56 D0 00131	20\$: MOVL STATUS, R0
			04 00134	RET

; Routine Size: 309 bytes, Routine Base: \$CODE\$ + 06A9


```

: 983 0978 1 %SBTTL ' SYSSMOD HOLDER - modify holder record'
: 984 0979 1 GLOBAL ROUTINE SYSSMOD_HOLDER (ID, HOLDER, SET_ATTRIB, CLR_ATTRIB) =
: 985 0980 1
: 986 0981 1 ++
: 987 0982 1
: 988 0983 1 FUNCTIONAL DESCRIPTION:
: 989 0984 1
: 990 0985 1 This routine modifies the specified holder record.
: 991 0986 1
: 992 0987 1 CALLING SEQUENCE:
: 993 0988 1 SYSSMOD_HOLDER (ID, HOLDER, SET_ATTRIB, CLR_ATTRIB)
: 994 0989 1
: 995 0990 1 INPUT PARAMETERS:
: 996 0991 1 ID: identifier longword
: 997 0992 1 HOLDER: address of the holder identifier quadword
: 998 0993 1 SET_ATTRIB: (optional) longword containing the attributes to set
: 999 0994 1 into the holder record
: 1000 0995 1 CLR_ATTRIB: (optional) longword containing the attributes to clear
: 1001 0996 1 in the holder record
: 1002 0997 1
: 1003 0998 1 IMPLICIT INPUTS:
: 1004 0999 1 NONE
: 1005 1000 1
: 1006 1001 1 OUTPUT PARAMETERS:
: 1007 1002 1 NONE
: 1008 1003 1
: 1009 1004 1 IMPLICIT OUTPUTS:
: 1010 1005 1 NONE
: 1011 1006 1
: 1012 1007 1 ROUTINE VALUE:
: 1013 1008 1 Status of operation
: 1014 1009 1
: 1015 1010 1 SIDE EFFECTS:
: 1016 1011 1 Holder record modified
: 1017 1012 1
: 1018 1013 1 --
: 1019 1014 1
: 1020 1015 2 BEGIN
: 1021 1016 2
: 1022 1017 2 LOCAL
: 1023 1018 2 LOC_ID : LONG, ! local copy of ID
: 1024 1019 2 LOC_HOLDER : REF VECTOR, ! local copy of HOLDER
: 1025 1020 2 HOLDER_ID : VECTOR [2], ! local copy of holder id quadword
: 1026 1021 2 LOC_SET_ATTRIB : LONG, ! local copy of SET_ATTRIB
: 1027 1022 2 LOC_CLR_ATTRIB : LONG, ! local copy of CLR_ATTRIB
: 1028 1023 2 ID_ATTRIB : LONG, ! attributes of identifier
: 1029 1024 2 STATUS : LONG, ! general status value
: 1030 1025 2 CLOSE : LONG, ! call to EXE$CLOSE_RDB required flag
: 1031 1026 2 RAB : $RAB DECL, ! RAB for file operations
: 1032 1027 2 REC_BUFFER : $BBLOCK [KGB$K_IDENT_RECORD]; ! general purpose record buffer
: 1033 1028 2
: 1034 1029 2
: 1035 1030 2
: 1036 1031 2 LABEL
: 1037 1032 2 RDB_OPEN; ! rights database is open in this block
: 1038 1033 2
: 1039 1034 2 ! Validate parameters
```

```
1040 1035 2 !
1041 1036 2
1042 1037 2 LOC_ID = .ID;
1043 1038 2 IF (.LOC_ID AND UIC$M_ID_FORM_FLAG) NEQU 0
1044 1039 2 THEN
1045 1040 2 (IF (.LOC_ID GTRU UIC$K_LAST_ID) THEN RETURN SS$_IVIDENT)
1046 1041 2 ELSE
1047 1042 2 (IF (.LOC_ID GTRU UIC$K_MAX_UIC) OR (.LOC_ID EQL 0) THEN RETURN SS$_IVIDENT);
1048 1043 2
1049 1044 2 LOC_HOLDER = .HOLDER;
1050 1045 2 IF NOT PROBER (%REF(0), %REF(8), .LOC_HOLDER) THEN RETURN SS$_ACCVIO;
1051 1046 2 HOLDER_ID[0] = .LOC_HOLDER[0];
1052 1047 2 HOLDER_ID[1] = .LOC_HOLDER[1];
1053 1048 2 IF .HOLDER_ID[0] GTRU UIC$K_MAX_UIC OR .HOLDER_ID[1] NEQU 0
1054 1049 2 THEN
1055 1050 2 RETURN SS$_IVIDENT;
1056 1051 2
1057 1052 2 LOC_SET_ATTRIB = .SET_ATTRIB;
1058 1053 2 IF (.LOC_SET_ATTRIB AND NOT KGB$M_VALID_ATTRIB) NEQU 0 THEN RETURN SS$_BADPARAM;
1059 1054 2
1060 1055 2 LOC_CLR_ATTRIB = .CLR_ATTRIB;
1061 1056 2 IF (.LOC_CLR_ATTRIB AND NOT KGB$M_VALID_ATTRIB) NEQU 0 THEN RETURN SS$_BADPARAM;
1062 1057 2
1063 1058 2 ! Get the rights database open for write.
1064 1059 2 !
1065 1060 2
1066 P 1061 2 $RAB_INIT (RAB = RAB,
1067 P 1062 2 RAC = KEY,
1068 P 1063 2 KRF = 0,
1069 P 1064 2 KBF = LOC_ID,
1070 P 1065 2 KSZ = 4,
1071 P 1066 2 ROP = (LIM, WAT, RLK, ULK),
1072 P 1067 2 UBF = REC_BUFFER,
1073 P 1068 2 USZ = KGB$K_IDENT_RECORD
1074 1069 2 );
1075 1070 2 STATUS = EXE$OPEN_RDB (0, 1, RAB[RAB$W_ISI], CLOSE);
1076 1071 2 IF NOT .STATUS THEN RETURN .STATUS;
1077 1072 2
1078 1073 2 RDB_OPEN:
1079 1074 2 BEGIN
1080 1075 2
1081 1076 2 ! Read and lock the ident record and save away its attributes.
1082 1077 2 !
1083 1078 2
1084 1079 2 STATUS = $GET (RAB = RAB);
1085 1080 2 IF .STATUS EQLU RMSS$_RNF THEN STATUS = SS$_NOSUCHID;
1086 1081 2 IF NOT .STATUS
1087 1082 2 THEN
1088 1083 2 BEGIN
1089 1084 2 $FREE (RAB = RAB);
1090 1085 2 LEAVE RDB_OPEN;
1091 1086 2 END;
1092 1087 2 ID_ATTRIB = .REC_BUFFER[KGB$L_ATTRIBUTES];
1093 1088 2
1094 1089 2 ! Read the holder records looking for the specified one.
1095 1090 2 !
1096 1091 2
```



```
: 1097      1092 3      RAB[RAB$V_ULK] = 0;
: 1098      1093 3      RAB[RAB$B_RAC] = RAB$C_SEQ;
: 1099      1094 3      WHILE 1 DO
: 1100      1095 4          BEGIN
: 1101      1096 4              STATUS = $GET (RAB = RAB);
: 1102      1097 4              IF .STATUS EQLU RMSS_EOF OR .STATUS EQLU RMSS_OK_LIM
: 1103      1098 4                  THEN
: 1104      1099 5                      BEGIN
: 1105      1100 5                          $FREE (RAB = RAB);
: 1106      1101 5                          STATUS = SSS_NOSUCHID;
: 1107      1102 5                          LEAVE RDB_OPEN;
: 1108      1103 4                      END;
: 1109      1104 4              IF CH$EQL (KGB$S_HOLDER, HOLDER_ID[0], KGB$S_HOLDER, REC_BUFFER[KGB$Q_HOLDER])
: 1110      1105 4                  THEN
: 1111      1106 4                      EXITLOOP;
: 1112      1107 4              END;
: 1113      1108 3
: 1114      1109 3      ! Now set and clear attributes as specified, but limited by the ident
: 1115      1110 3      ! record attributes.
: 1116      1111 3      !
: 1117      1112 3
: 1118      1113 3      IF .LOC_CLR_ATTRIB NEQU 0
: 1119      1114 3      THEN
: 1120      1115 3          REC_BUFFER[KGB$L_ATTRIBUTES] =
: 1121      1116 3              .REC_BUFFER[KGB$C_ATTRIBUTES] AND NOT .LOC_CLR_ATTRIB;
: 1122      1117 3
: 1123      1118 3      IF .LOC_SET_ATTRIB NEQU 0
: 1124      1119 3      THEN
: 1125      1120 3          REC_BUFFER[KGB$L_ATTRIBUTES] =
: 1126      1121 3              (.REC_BUFFER[KGB$L_ATTRIBUTES] OR .LOC_SET_ATTRIB) AND .ID_ATTRIB;
: 1127      1122 3
: 1128      1123 3      STATUS = $UPDATE (RAB = RAB);
: 1129      1124 3      $FREE (RAB = RAB);
: 1130      1125 3      END;
: 1131      1126 2
: 1132      1127 2      ! Close the rights database if there is no image
: 1133      1128 2      !
: 1134      1129 2
: 1135      1130 2      IF .CLOSE THEN EXE$CLOSE_RDB();
: 1136      1131 2      IF .STATUS
: 1137      1132 2      THEN
: 1138      1133 2          RETURN SSS_NORMAL
: 1139      1134 2      ELSE
: 1140      1135 2          RETURN .STATUS;
: 1141      1136 2
: 1142      1137 1      END;                                     ! End of routine SYSSMOD_HOLDER
```

```
03FC 00000
59 00000000G 00 9E 00002
58 00000000G 00 9E 00009
5E          80 AE 9E 00010
           04 AC DD 00014
```

```
.ENTRY SYSSMOD_HOLDER, Save R2,R3,R4,R5,R6,R7,R8,- : 0979
R9
MOVAB SYSSFREE, R9
MOVAB SYSSGET, R8
MOVAB -128(SP), SP
PUSHL ID : 1037
```

	8FFFFFFF	8F		0B 18 00017	BGEQ	1\$		1038
				6E D1 00019	CMPL	LOC_ID, #-1879048193		1040
				0F 1B 00020	BLEQU	2\$		
	3FFFFFFF	8F		33 11 00022	BRB	4\$		
				6E D1 00024	CMPL	LOC_ID, #1073741823		1042
				2A 1A 0002B	BGTRU	4\$		
				6E D5 0002D	TSTL	LOC_ID		
				26 13 0002F	BEQL	4\$		
		50	08	AC D0 00031	MOVL	HOLDER, LOC_HOLDER		1044
60		08		00 0C 00035	PROBER	#0, #8, (LOC_HOLDER)		1045
				04 12 00039	BNEQ	3\$		
		50		0C D0 0003B	MOVL	#12, R0		
				04 0003E	RET			
	7C AE			60 D0 0003F	MOVL	(LOC_HOLDER), HOLDER_ID		1046
	FC AD	04		A0 D0 00043	MOVL	4(LOC_HOLDER), HOLDER_ID+4		1047
	3FFFFFFF	8F	7C	AE D1 00048	CMPL	HOLDER_ID, #1073741823		1048
				05 1A 00050	BGTRU	4\$		
			FC	AD D5 00052	TSTL	HOLDER_ID+4		
				06 13 00055	BEQL	5\$		
		50	2224	8F 3C 00057	MOVZWL	#8740, R0		1050
				04 0005C	RET			
		57	0C	AC D0 0005D	MOVL	SET_ATTRIB, LOC_SET_ATTRIB		1052
	FFFFFFFFE	8F		57 D3 00061	BITL	LOC_SET_ATTRIB, #-2		1053
				0D 12 00068	BNEQ	6\$		
		56	10	AC D0 0006A	MOVL	CLR_ATTRIB, LOC_CLR_ATTRIB		1055
	FFFFFFFFE	8F		56 D3 0006E	BITL	LOC_CLR_ATTRIB, #-2		1056
				04 13 00075	BEQL	7\$		
		50		14 D0 00077	MOVL	#20, R0		
				04 0007A	RET			
0044	8F	00		00 2C 0007B	MOVCS	#0, (SP), #0, #68, \$RMS_PTR		1069
			38	AE 00082				
	38 AE	4401		8F B0 00084	MOVW	#17409, \$RMS_PTR		
	3C AE	000E4000		8F D0 0008A	MOVL	#933888, \$RMS_PTR+4		
	56 AE			01 90 00092	MOVB	#1, \$RMS_PTR+30		
	58 AE			30 B0 00096	MOVW	#48, \$RMS_PTR+32		
	5C AE	08		AE 9E 0009A	MOVAB	REC_BUFFER, \$RMS_PTR+36		
	68 AE			6E 9E 0009F	MOVAB	LOC_ID, \$RMS_PTR+48		
	6C AE			04 90 000A3	MOVB	#4, \$RMS_PTR+52		
			04	AE 9F 000A7	PUSHAB	CLOSE		
			3E	AE 9F 000AA	PUSHAB	RAB+2		1070
				01 DD 000AD	PUSHL	#1		
				7E D4 000AF	CLRL	-(SP)		
	00000000G	9F		04 FB 000B1	CALLS	#4, @#EXESOPEN_RDB		
		54		50 D0 000B8	MOVL	R0, STATUS		
		03		54 E8 000BB	BLBS	STATUS, 8\$		1071
			0093	31 000BE	BRW	18\$		
			38	AE 9F 000C1	PUSHAB	RAB		1079
		68		01 FB 000C4	CALLS	#1, SYSSGET		
		54		50 D0 000C7	MOVL	R0, STATUS		
	000182B2	8F		54 D1 000CA	CMPL	STATUS, #98994		1080
				05 12 000D1	BNEQ	9\$		
		54	21EC	8F 3C 000D3	MOVZWL	#8684, STATUS		
		61		54 E9 000D8	BLBC	STATUS, 15\$		1081
		55	0C	AE D0 000DB	MOVL	REC_BUFFER+4, ID_ATTRIB		1087
	3E	AE		04 8A 000DF	BICB2	#4, -RAB+6		1092
			56	AE 94 000E3	CLRB	RAB+30		1093
			38	AE 9F 000E6	PUSHAB	RAB		1096

		68	01	FB	000E9	CALLS	#1, SYSSGET	:	
		54	50	D0	000EC	MOVL	R0, STATUS	:	
	0001827A	8F	54	D1	000EF	CMPL	STATUS, #98938	:	1097
			09	13	000F6	BEQL	11\$:	
	00018051	8F	54	D1	000F8	CMPL	STATUS, #98385	:	
			0D	12	000FF	BNEQ	12\$:	
			38	AE	9F 00101	PUSHAB	RAB	:	1100
		69	01	FB	00104	CALLS	#1, SYSSFREE	:	
		54	8F	3C	00107	MOVZWL	#8684, STATUS	:	1101
			34	11	0010C	BRB	16\$:	1102
10	AE	7C	AE	08	29 0010E	CMPC3	#8, HOLDER_ID, REC_BUFFER+8	:	1105
				D0	12 00114	BNEQ	10\$:	
				56	D5 00116	TSTL	LOC_CLR_ATTRIB	:	1114
				04	13 00118	BEQL	13\$:	
		OC	AE	56	CA 0011A	BICL2	LOC_CLR_ATTRIB, REC_BUFFER+4	:	1117
				57	D5 0011E	TSTL	LOC_SET_ATTRIB	:	1118
				0D	13 00120	BEQL	14\$:	
	50	OC	AE	57	C9 00122	BISL3	LOC_SET_ATTRIB, REC_BUFFER+4, R0	:	1121
				55	D2 00127	MCOML	ID_ATTRIB, R1	:	
OC	AE			51	CB 0012A	BICL3	R1, R0, REC_BUFFER+4	:	
			38	AE	9F 0012F	PUSHAB	RAB	:	1123
	00000000G	00	01	FB	00132	CALLS	#1, SYSSUPDATE	:	
		54	50	D0	00139	MOVL	R0, STATUS	:	
			38	AE	9F 0013C	PUSHAB	RAB	:	1124
		69	01	FB	0013F	CALLS	#1, SYSSFREE	:	
		07	AE	E9 00142	BLBC	CLOSE, 17\$:	1130
	00000000G	9F	00	FB	00146	CALLS	#0, @#EXE\$CLOSE_RDB	:	
		04	54	E9 0014D	BLBC	STATUS, 18\$:	1131
		50	01	D0	00150	MOVL	#1, R0	:	1135
				04	00153	RET		:	
		50	54	D0	00154	MOVL	STATUS, R0	:	
				04	00157	RET		:	1137

; Routine Size: 344 bytes, Routine Base: \$CODE\$ + 07DE

; 1143 1138 1

```
: 1145 1139 1 %SBTTL ' SYSSMOD_IDENT - Modify identifier record'
: 1146 1140 1 GLOBAL ROUTINE SYSSMOD_IDENT (ID, SET_ATTRIB, CLR_ATTRIB, NEW_NAME, NEW_ID) =
: 1147 1141 1
: 1148 1142 1 ++
: 1149 1143 1
: 1150 1144 1 FUNCTIONAL DESCRIPTION:
: 1151 1145 1
: 1152 1146 1     This routine modifies the attributes of the specified
: 1153 1147 1     identifier.
: 1154 1148 1
: 1155 1149 1 CALLING SEQUENCE:
: 1156 1150 1     SYSSMOD_IDENT (ID, SET_ATTRIB, CLR_ATTRIB, NEW_NAME, NEW_ID )
: 1157 1151 1
: 1158 1152 1 INPUT PARAMETERS:
: 1159 1153 1     ID: identifier longword
: 1160 1154 1     SET_ATTRIB: (optional) longword containing the attributes
: 1161 1155 1     to set into the identifier record
: 1162 1156 1     CLR_ATTRIB: (optional) longword containing the attributes
: 1163 1157 1     to clear in the identifier record
: 1164 1158 1     NEW_NAME: address of a character string descriptor for
: 1165 1159 1     the new name
: 1166 1160 1     NEW_ID: new identifier value
: 1167 1161 1
: 1168 1162 1 IMPLICIT INPUTS:
: 1169 1163 1     NONE
: 1170 1164 1
: 1171 1165 1 OUTPUT PARAMETERS:
: 1172 1166 1     NONE
: 1173 1167 1
: 1174 1168 1 IMPLICIT OUTPUTS:
: 1175 1169 1     NONE
: 1176 1170 1
: 1177 1171 1 ROUTINE VALUE:
: 1178 1172 1     Status of operation
: 1179 1173 1
: 1180 1174 1 SIDE EFFECTS:
: 1181 1175 1     Identifier record modified
: 1182 1176 1
: 1183 1177 1 --
: 1184 1178 1
: 1185 1179 2 BEGIN
: 1186 1180 2 LITERAL
: 1187 1181 2     BUFFER_LENGTH = MAX ( KGB$K_HOLD_RECORD,
: 1188 1182 2     KGB$K_IDENT_RECORD,
: 1189 1183 2     KGB$K_MAINT_RECORD ) ;
: 1190 1184 2
: 1191 1185 2 LOCAL
: 1192 1186 2     LOC_ID : $BBLOCK[4],      ! local copy of ID
: 1193 1187 2     LOC_SET_ATTRIB : LONG,      ! local copy of SET_ATTRIB
: 1194 1188 2     LOC_CLR_ATTRIB : LONG,      ! local copy of CLR_ATTRIB
: 1195 1189 2     LOC_NEW_NAME : LONG,      ! local copy of NEW_NAME
: 1196 1190 2     LOC_NEW_ID : $BBLOCK[4],   ! local copy of NEW_ID
: 1197 1191 2     NEW_NAME_LEN : LONG,      ! Length of new name
: 1198 1192 2     NEW_NAMEADR : LONG,      ! address of new name
: 1199 1193 2     STATUS : LONG,          ! general status value
: 1200 1194 2     CLOSE : LONG,          ! call to EXE$CLOSE_RDB required flag
: 1201 1195 2     RAB : $RAB_DECL,      ! RAB for file I/O
```



```
: 1202      1196 2      REC_BUFFER      : $BLOCK [BUFFER_LENGTH];
: 1203      1197 2                                     ! record buffer for records
: 1204      1198 2
: 1205      1199 2 LABEL
: 1206      1200 2      RDB_OPEN;                                     ! rights database is open in this block
: 1207      1201 2
: 1208      1202 2      ! Validate parameters
: 1209      1203 2      !
: 1210      1204 2
: 1211      1205 2      LOC_ID = .ID;
: 1212      1206 2      IF .LOC_ID[UIC$V_FORMAT] EQL UIC$K_ID_FORMAT
: 1213      1207 2      THEN
: 1214      1208 2          (IF (.LOC_ID GTRU UIC$K_LAST_ID) THEN RETURN SSS_IVIDENT)
: 1215      1209 2      ELSE
: 1216      1210 2          (IF (.LOC_ID GTRU UIC$K_MAX_UIC) OR (.LOC_ID EQL 0) THEN RETURN SSS_IVIDENT);
: 1217      1211 2
: 1218      1212 2      LOC_SET_ATTRIB = .SET_ATTRIB;
: 1219      1213 2      IF T.LOC_SET_ATTRIB AND NOT KGB$M_VALID_ATTRIB) NEQU 0 THEN RETURN SSS_BADPARAM;
: 1220      1214 2
: 1221      1215 2      LOC_CLR_ATTRIB = .CLR_ATTRIB;
: 1222      1216 2      IF T.LOC_CLR_ATTRIB AND NOT KGB$M_VALID_ATTRIB) NEQU 0 THEN RETURN SSS_BADPARAM;
: 1223      1217 2
: 1224      1218 2      LOC_NEW_NAME = .NEW_NAME ;
: 1225      1219 2      IF .LOC_NEW_NAME NEQ 0
: 1226      1220 2      THEN
: 1227      1221 2          BEGIN
: 1228      1222 2              STATUS = EXESVAL_IDNAME ( .LOC_NEW_NAME ; NEW_NAMLEN, NEW_NAMADR ) ;
: 1229      1223 2              IF NOT .STATUS THEN RETURN .STATUS;
: 1230      1224 2              END ;
: 1231      1225 2
: 1232      1226 2      LOC_NEW_ID = .NEW_ID;
: 1233      1227 2      IF .LOC_NEW_ID NEQ 0
: 1234      1228 2      THEN
: 1235      1229 2          BEGIN
: 1236      1230 2              IF .LOC_NEW_ID[UIC$V_FORMAT] EQL UIC$K_ID_FORMAT
: 1237      1231 2              THEN
: 1238      1232 2                  (IF (.LOC_NEW_ID GTRU UIC$K_LAST_ID) THEN RETURN SSS_IVIDENT)
: 1239      1233 2              ELSE
: 1240      1234 2                  (IF (.LOC_NEW_ID GTRU UIC$K_MAX_UIC) THEN RETURN SSS_IVIDENT);
: 1241      1235 2
: 1242      1236 2              ! Do not allow a format switch
: 1243      1237 2              !
: 1244      1238 2              IF .LOC_ID[UIC$V_FORMAT] NEQ .LOC_NEW_ID[UIC$V_FORMAT]
: 1245      1239 2              THEN RETURN SSS_IVIDENT;
: 1246      1240 2              END ;
: 1247      1241 2
: 1248      1242 2      ! Open the rights database for writing.
: 1249      1243 2      !
: 1250      P 1244 2      $RAB_INIT (RAB = RAB,
: 1251      P 1245 2          RAC = KEY,
: 1252      P 1246 2          KRF = 0,
: 1253      P 1247 2          KSZ = 4,
: 1254      P 1248 2          KBF = LOC_ID,
: 1255      P 1249 2          ROP = (LIM, WAT, RLK, ULK),
: 1256      P 1250 2          USZ = BUFFER_LENGTH,
: 1257      P 1251 2          UBF = REC_BUFFER
: 1258      1252 2          );
```

```
1259 1253 2 STATUS = EXESOPEN RDB (0, 1, RAB[RAB$W_ISI], CLOSE);
1260 1254 2 IF NOT .STATUS THEN RETURN .STATUS;
1261 1255 2
1262 1256 2 RDB_OPEN:
1263 1257 2 BEGIN
1264 1258 2
1265 1259 2 | Modify the identifier name
1266 1260 2 |
1267 1261 2 | IF .LOC_NEW_NAME NEQ 0
1268 1262 2 | THEN
1269 1263 2 | BEGIN
1270 1264 2 |     STATUS = SYSSMOD_IDENT_NAME ( RAB, .LOC_ID, .NEW_NAMLEN, .NEW_NAMADR ) ;
1271 1265 2 |     IF NOT .STATUS THEN LEAVE RDB_OPEN ;
1272 1266 2 |     END ;
1273 1267 2 |
1274 1268 2 | Modify the identifier attributes
1275 1269 2 |
1276 1270 2 | IF ( .LOC_CLR_ATTRIB NEQ 0 ) OR
1277 1271 2 | ( .LOC_SET_ATTRIB NEQ 0 )
1278 1272 2 | THEN
1279 1273 2 | BEGIN
1280 1274 2 |     STATUS = SYSSMOD_IDENT_ATTRIB ( RAB, .LOC_ID,
1281 1275 2 |     .LOC_SET_ATTRIB, .LOC_CLR_ATTRIB ) ;
1282 1276 2 |     IF NOT .STATUS THEN LEAVE RDB_OPEN ;
1283 1277 2 |     END ;
1284 1278 2 |
1285 1279 2 | Modify the identifier value
1286 1280 2 |
1287 1281 2 | IF .LOC_NEW_ID NEQ 0
1288 1282 2 | THEN
1289 1283 2 | BEGIN
1290 1284 2 |     STATUS = SYSSMOD_IDENT_ID ( RAB, .LOC_ID, .LOC_NEW_ID ) ;
1291 1285 2 |     IF NOT .STATUS THEN LEAVE RDB_OPEN ;
1292 1286 2 |     END ;
1293 1287 2 |
1294 1288 2 | END; ! End of RDB_OPEN
1295 1289 2 |
1296 1290 2 | Close the rights database if there is no image
1297 1291 2 |
1298 1292 2 | IF .CLOSE THEN EXESCLOSE_RDB();
1299 1293 2 | IF .STATUS
1300 1294 2 | THEN
1301 1295 2 |     RETURN SSS_NORMAL
1302 1296 2 | ELSE
1303 1297 2 |     RETURN .STATUS;
1304 1298 2 |
1305 1299 2 |
1306 1300 2 |
1307 1301 2 |
1308 1302 2 |
1309 1303 2 |
1310 1304 1 END; ! End of routine SYSSMOD_IDENT
```


				OFFC 00000	.ENTRY	SYSSMOD_IDENT, Save R2,R3,R4,R5,R6,R7,R8,-	
			5E	FF68	CE 9E 00002	R9,R10,R11	1140
		OC	AE	04	AC D0 00007	-152(SP), SP	
			57	OC	AE D0 0000C	ID, LOC_ID	1205
02	OF	AE	02		06 ED 00010	LOC_ID, R7	1208
		8FFFFFFF	8F		0B 12 00016	#6, #2, LOC_ID+3, #2	1206
					57 D1 00018	1\$	
					0F 1B 0001F	R7, #-1879048193	1208
		3FFFFFFF	8F		7D 11 00021	2\$	
					57 D1 00023 1\$:	R7, #1073741823	1210
					74 1A 0002A	8\$	
					57 D5 0002C	R7	
					70 13 0002E	8\$	
		5A	08		AC D0 00030 2\$:	SET_ATTRIB, LOC_SET_ATTRIB	1212
		FFFFFFFE	8F		5A D3 00034	LOC_SET_ATTRIB, #-2	1213
					0D 12 0003B	3\$	
					AC D0 0003D	CLR_ATTRIB, LOC_CLR_ATTRIB	1215
		59	OC		59 D3 00041	LOC_CLR_ATTRIB, #-2	1216
		FFFFFFFE	8F		04 13 00048	4\$	
					14 D0 0004A 3\$:	#20, R0	
					04 0004D		
					AC D0 0004E 4\$:	NEW_NAME, LOC_NEW_NAME	1218
					6E D4 00052	(SP)	1219
					51 D5 00054	LOC_NEW_NAME	
					16 13 00056	5\$	
					6E D6 00058	(SP)	
					9F 16 0005A	@#EXESVAL_IDNAME	1222
		00000000G			50 D0 00060	R0, STATUS	
					51 D0 00063	R1, 8(SP)	
		08			52 D0 00067	R2, 4(SP)	
		04			56 E9 0006B	STATUS, 10\$	1223
					AC D0 0006E 5\$:	NEW_ID, LOC_NEW_ID	1226
					5B D4 00072	R11	1227
					58 D5 00074	LOC_NEW_ID	
					2E 13 00076	9\$	
					5B D6 00078	R11	
02		58	02		1E ED 0007A	#30, #2, LOC_NEW_ID, #2	1230
					09 12 0007F	6\$	
		8FFFFFFF	8F		58 D1 00081	LOC_NEW_ID, #-1879048193	1232
					07 11 00088	7\$	
		3FFFFFFF	8F		58 D1 0008A 6\$:	LOC_NEW_ID, #1073741823	1234
					0D 1A 00091 7\$:	8\$	
50					1E EF 00093	#30, #2, LOC_NEW_ID, R0	1238
50	OF	58	02		06 ED 00098	#6, #2, LOC_ID+3, R0	
		AE	02		06 13 0009E	9\$	
					8F 3C 000A0 8\$:	#8740, R0	1239
					04 000A5		
0044	8F	00	6E		00 2C 000A6 9\$:	#0, (SP), #0, #68, \$RMS_PTR	1252
					AE 000AD		
		54			8F B0 000AF	#17409, \$RMS_PTR	
		58			8F D0 000B5	#933888, \$RMS_PTR+4	
		72			01 90 000BD	#1, \$RMS_PTR+30	
		74			8F 9B 000C1	#64, \$RMS_PTR+32	
		78			AE 9E 000C6	REC_BUFFER, \$RMS_PTR+36	
		EC			AE 9E 000CB	LOC_ID, \$RMS_PTR+48	
		FO			04 90 000D0	#4, \$RMS_PTR+52	

RDBSHR
V04-000

RDBSHR - Rights database loadable system service
SYSSMOD_IDENT - Modify identifier record

J 10
16-Sep-1984 01:48:50
14-Sep-1984 12:40:52

VAX-11 Bliss-32 V4.0-742
[LOADSS.SRC]RDBSHR.B32;1

Page 40
(7)

		10	AE	9F	000D4	PUSHAB	CLOSE	:	1253
		5A	AE	9F	000D7	PUSHAB	RAB+2	:	
			01	DD	000DA	PUSHL	#1	:	
			7E	D4	000DC	CLRL	-(SP)	:	
00000000G	9F		04	FB	000DE	CALLS	#4, @#EXESOPEN_RDB	:	
	56		50	D0	000E5	MOVL	R0, STATUS	:	
	58		56	E9	000E8	10\$:	BLBC	STATUS, 16\$	1254
	16		6E	E9	000EB	BLBC	(SP), 11\$:	1262
		04	AE	DD	000EE	PUSHL	NEW_NAMADR	:	1265
		0C	AE	DD	000F1	PUSHL	NEW_NAMLEN	:	
			57	DD	000F4	PUSHL	R7	:	
0000V	CF	60	AE	9F	000F6	PUSHAB	RAB	:	
	56		04	FB	000F9	CALLS	#4, SYSSMOD_IDENT_NAME	:	
	2D		50	D0	000FE	MOVL	R0, STATUS	:	
			56	E9	00101	BLBC	STATUS, 14\$:	1266
			59	D5	00104	11\$:	TSTL	LOC_CLR_ATTRIB	1272
			04	12	00106	BNEQ	12\$:	
			5A	D5	00108	TSTL	LOC_SET_ATTRIB	:	1273
			14	13	0010A	BEQL	13\$:	
			59	DD	0010C	12\$:	PUSHL	LOC CLR ATTRIB	1277
		0480	8F	BB	0010E	PUSHR	#*MZR7,R10>	:	1276
		60	AE	9F	00112	PUSHAB	RAB	:	
0000V	CF		04	FB	00115	CALLS	#4, SYSSMOD_IDENT_ATTRIB	:	
	56		50	D0	0011A	MOVL	R0, STATUS	:	
	11		56	E9	0011D	BLBC	STATUS, 14\$:	1278
	0E		5B	E9	00120	13\$:	BLBC	R11, 14\$	1285
	7E		57	7D	00123	MOVQ	R7, -(SP)	:	1288
		5C	AE	9F	00126	PUSHAB	RAB	:	
0000V	CF		03	FB	00129	CALLS	#3, SYSSMOD_IDENT_ID	:	
	56		50	D0	0012E	MOVL	R0, STATUS	:	
	07		AE	E9	00131	14\$:	BLBC	CLOSE, 15\$	1297
00000000G	9F	10	00	FB	00135	CALLS	#0, @#EXESCLOSE_RDB	:	
	04		56	E9	0013C	15\$:	BLBC	STATUS, 16\$	1298
	50		01	D0	0013F	MOVL	#1, R0	:	1302
				04	00142	RET		:	
	50		56	D0	00143	16\$:	MOVL	STATUS, R0	
			04	00146	RET			:	1304

; Routine Size: 327 bytes, Routine Base: \$CODE\$ + 0936

; 1311 1305 1


```
: 1313 1306 1 %SBTTL ' SYSSMOD_IDENT_ATTRIB - Modify identifier attributes'
: 1314 1307 1 ROUTINE SYSSMOD_IDENT_ATTRIB ( RAB_PTR, ID, SET_ATTRIB, CLR_ATTRIB) =
: 1315 1308 1
: 1316 1309 1 ++
: 1317 1310 1
: 1318 1311 1 FUNCTIONAL DESCRIPTION:
: 1319 1312 1
: 1320 1313 1 This routine modifies the attributes of the specified
: 1321 1314 1 identifier.
: 1322 1315 1
: 1323 1316 1 CALLING SEQUENCE:
: 1324 1317 1 SYSSMOD_IDENT_ATTRIB ( RAB_PTR, ID, SET_ATTRIB, CLR_ATTRIB)
: 1325 1318 1
: 1326 1319 1 INPUT PARAMETERS:
: 1327 1320 1 RAB_PTR: address of RAB for open rights data base file
: 1328 1321 1 ID: identifier longword
: 1329 1322 1 SET_ATTRIB: (optional) longword containing the attributes
: 1330 1323 1 to set into the identifier record
: 1331 1324 1 CLR_ATTRIB: (optional) longword containing the attributes
: 1332 1325 1 to clear in the identifier record
: 1333 1326 1
: 1334 1327 1 IMPLICIT INPUTS:
: 1335 1328 1 NONE
: 1336 1329 1
: 1337 1330 1 OUTPUT PARAMETERS:
: 1338 1331 1 NONE
: 1339 1332 1
: 1340 1333 1 IMPLICIT OUTPUTS:
: 1341 1334 1 NONE
: 1342 1335 1
: 1343 1336 1 ROUTINE VALUE:
: 1344 1337 1 Status of operation
: 1345 1338 1
: 1346 1339 1 SIDE EFFECTS:
: 1347 1340 1 Identifier record modified
: 1348 1341 1
: 1349 1342 1 --
: 1350 1343 1
: 1351 1344 2 BEGIN
: 1352 1345 2
: 1353 1346 2 LABEL
: 1354 1347 2 MOD_ATTRIB ;
: 1355 1348 2
: 1356 1349 2 BIND
: 1357 1350 2 RAB = .RAB_PTR : $RAB_DECL ,
: 1358 1351 2 REC_BUFFER = .RAB[RAB$L_UBF] : $BBLOCK ;
: 1359 1352 2
: 1360 1353 2 LOCAL
: 1361 1354 2 KRFSAV : BYTE ,
: 1362 1355 2 KSZSAV : BYTE ,
: 1363 1356 2 KBFSAV : LONG ,
: 1364 1357 2 RACSAV : BYTE ,
: 1365 1358 2 OPSAV : LONG ,
: 1366 1359 2 USZSAV : WORD ,
: 1367 1360 2 IDENT_RFA : $BBLOCK [RAB$$_RFA], ! RFA of ident record
: 1368 1361 2 STATUS : LONG ;
: 1369 1362 2
```

```
1370 1363 2 |
1371 1364 2 | Save the state of the RAB
1372 1365 2 |
1373 1366 2 KRFSAV = .RAB[RAB$B_KRF] ;
1374 1367 2 KSZSAV = .RAB[RAB$B_KSZ] ;
1375 1368 2 KBFSAV = .RAB[RAB$B_KBF] ;
1376 1369 2 RACSAV = .RAB[RAB$B_RAC] ;
1377 1370 2 OPSAV = .RAB[RAB$B_ROP] ;
1378 1371 2 USZSAV = .RAB[RAB$W_USZ] ;
1379 1372 2 |
1380 1373 2 |
1381 1374 2 | Set up the RAB for key record access using the id key (primary)
1382 1375 2 |
1383 1376 2 RAB[RAB$B_RAC] = RAB$C_KEY ;
1384 1377 2 RAB[RAB$B_KBF] = ID ;
1385 1378 2 RAB[RAB$B_KSZ] = 4 ;
1386 1379 2 RAB[RAB$B_KRF] = 0 ;
1387 1380 2 RAB[RAB$B_ROP] = RAB$M_LIM OR
1388 1381 2 RAB$M_WAT OR
1389 1382 2 RAB$M_RLK OR
1390 1383 2 RAB$M_ULK ;
1391 1384 2 RAB[RAB$W_USZ] = KGB$K_IDENT_RECORD ;
1392 1385 2 |
1393 1386 2 MOD_ATTRIB:
1394 1387 2 BEGIN
1395 1388 2 |
1396 1389 2 | If we are clearing attributes, we have to fix up the holder records
1397 1390 2 | first. Locate the identifier record.
1398 1391 2 |
1399 1392 2 IF .CLR_ATTRIB NEQU 0
1400 1393 2 THEN
1401 1394 2 BEGIN
1402 1395 2 STATUS = $GET (RAB = RAB);
1403 1396 2 IF .STATUS EQLU RMSS_RNF THEN STATUS = SSS_NOSUCHID;
1404 1397 2 IF NOT .STATUS THEN LEAVE MOD_ATTRIB ;
1405 1398 2 CH$MOVE (RAB$S_RFA, RAB[RAB$W_RFA], IDENT_RFA);
1406 1399 2 |
1407 1400 2 | Now sequentially locate all the holder records and modify them.
1408 1401 2 |
1409 1402 2 |
1410 1403 2 RAB[RAB$B_RAC] = RAB$C_SEQ;
1411 1404 2 RAB[RAB$V_ULK] = 0;
1412 1405 2 WHILE 1 DO
1413 1406 2 BEGIN
1414 1407 2 STATUS = $GET (RAB = RAB);
1415 1408 2 IF .STATUS EQLU RMSS_EOF OR .STATUS EQLU RMSS_OK_LIM THEN EXITLOOP;
1416 1409 2 IF NOT .STATUS THEN LEAVE MOD_ATTRIB ;
1417 1410 2 |
1418 1411 2 REC_BUFFER[KGB$L_ATTRIBUTES] =
1419 1412 2 .REC_BUFFER[KGB$C_ATTRIBUTES] AND NOT .CLR_ATTRIB;
1420 1413 2 STATUS = $UPDATE (RAB = RAB);
1421 1414 2 IF NOT .STATUS THEN LEAVE MOD_ATTRIB ;
1422 1415 2 END;
1423 1416 2 |
1424 1417 2 RAB[RAB$B_RAC] = RAB$C_RFA;
1425 1418 2 CH$MOVE (RAB$S_RFA, IDENT_RFA, RAB[RAB$W_RFA]);
1426 1419 2 END;
```



```

1427      ! Read the ident record, set and clear attributes as directed, and write
1428      ! it back.
1429
1430      STATUS = $GET (RAB = RAB);
1431      IF .STATUS EQLU RMSS RNF THEN STATUS = SSS_NOSUCHID;
1432      IF NOT .STATUS THEN LEAVE MOD_ATTRIB ;
1433
1434      IF .CLR_ATTRIB NEQU 0
1435      THEN
1436          REC_BUFFER[KGB$L_ATTRIBUTES] =
1437          .REC_BUFFER[KGB$[_ATTRIBUTES] AND NOT .CLR_ATTRIB;
1438      IF .SET_ATTRIB NEQU 0
1439      THEN
1440          REC_BUFFER[KGB$L_ATTRIBUTES] =
1441          .REC_BUFFER[KGB$[_ATTRIBUTES] OR .SET_ATTRIB;
1442      STATUS = $UPDATE (RAB = RAB);
1443      END;
1444
1445      ! Clean up locks.
1446      $FREE ( RAB = RAB ) ;
1447
1448      ! Restore RAB
1449      RAB[RAB$B_KRF] = .KRFSAV ;
1450      RAB[RAB$B_KSZ] = .KSZSAV ;
1451      RAB[RAB$L_KBF] = .KBFSAV ;
1452      RAB[RAB$B_RAC] = .RACSAV ;
1453      RAB[RAB$L_ROP] = .ROPSAV ;
1454      RAB[RAB$W_USZ] = .USZSAV ;
1455
1456      ! Get back to the beginning
1457      IF .STATUS
1458      THEN STATUS = $REWIND ( RAB = RAB ) ;
1459
1460      RETURN .STATUS;
1461
1462      ! End of routine SYSSMOD_IDENT_ATTRIB
1463      END;
1464

```

```
! End of routine SYSSMOD_IDENT_ATTRIB
```

.EXTRN SYSS\$REWIND									
OFFC 00000 SYSS\$MOD_IDENT_ATTRIB:									
						.WORD	Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11		: 1307
						SUBL2	#32, SP		:
	5E	20	C2	00002		MOVL	RAB PTR, R6		: 1350
	56	04	AC	D0	00005	MOVL	36(R6), R7		: 1351
	57	24	A6	D0	00009	MOVB	53(R6), KRFSAV		: 1366
14	AE	35	A6	90	0000D	MOVB	52(R6), KSZSAV		: 1367
10	AE	34	A6	90	00012				

	0C	AE	30	A6	D0	00017	MOVL	48(R6), KBFSAV	1368
		5A	1E	A6	9E	0001C	MOVAB	30(R6), R10	1369
	08	AE		6A	90	00020	MOVB	(R10), RACSAV	
	04	AE	04	A6	D0	00024	MOVL	4(R6), ROPSAV	1370
		6E	20	A6	B0	00029	MOVW	32(R6), USZSAV	1371
		6A		01	90	0002D	MOVB	#1, (R10)	1376
	30	A6	08	AC	9E	00030	MOVAB	ID, 48(R6)	1377
	34	A6		04	B0	00035	MOVW	#4, 52(R6)	1378
	04	A6	000E4000	8F	D0	00039	MOVL	#933888, 4(R6)	1382
	20	A6		30	B0	00041	MOVW	#48, 32(R6)	1384
		59	10	AC	D0	00045	MOVL	CLR_ATTRIB, R9	1392
				5B	D4	00049	CLRL	R11	
				59	D5	0004B	TSTL	R9	
				6A	13	0004D	BEQL	4\$	
				5B	D6	0004F	INCL	R11	
				56	DD	00051	PUSHL	R6	1395
	00000000G	00		01	FB	00053	CALLS	#1, SYSSGET	
		58		50	D0	0005A	MOVL	R0, STATUS	
	000182B2	8F		58	D1	0005D	CMPL	STATUS, #98994	1396
				05	12	00064	BNEQ	1\$	
		58	21EC	8F	3C	00066	MOVZWL	#8684, STATUS	
		65		58	E9	0006B	BLBC	STATUS, 5\$	1397
18	AE	10		06	28	0006E	MOVCL	#6, 16(R6), IDENT_RFA	1398
				6A	94	00074	CLRB	(R10)	1403
		06		04	8A	00076	BICB2	#4, 6(R6)	1404
				56	DD	0007A	PUSHL	R6	1407
	00000000G	00		01	FB	0007C	CALLS	#1, SYSSGET	
		58		50	D0	00083	MOVL	R0, STATUS	
	0001827A	8F		58	D1	00086	CMPL	STATUS, #98938	1408
				21	13	0008D	BEQL	3\$	
	00018051	8F		58	D1	0008F	CMPL	STATUS, #98385	
				18	13	00096	BEQL	3\$	
		58		58	E9	00098	BLBC	STATUS, 8\$	1409
		04		59	CA	0009B	BICL2	R9, 4(R7)	1412
				56	DD	0009F	PUSHL	R6	1413
	00000000G	00		01	FB	000A1	CALLS	#1, SYSSUPDATE	
		58		50	D0	000A8	MOVL	R0, STATUS	
		CC		58	E8	000AB	BLBS	STATUS, 2\$	1414
				43	11	000AE	BRB	8\$	
		6A		02	90	000B0	MOVB	#2, (R10)	1417
10	A6	18		06	28	000B3	MOVCL	#6, IDENT_RFA, 16(R6)	1418
				56	DD	000B9	PUSHL	R6	1425
	00000000G	00		01	FB	000BB	CALLS	#1, SYSSGET	
		58		50	D0	000C2	MOVL	R0, STATUS	
	000182B2	8F		58	D1	000C5	CMPL	STATUS, #98994	1426
				05	12	000CC	BNEQ	5\$	
		58	21EC	8F	3C	000CE	MOVZWL	#8684, STATUS	
		1D		58	E9	000D3	BLBC	STATUS, 8\$	1427
		04		5B	E9	000D6	BLBC	R11, 6\$	1429
		04		59	CA	000D9	BICL2	R9, 4(R7)	1432
			0C	AC	D5	000DD	TSTL	SET_ATTRIB	1433
				05	13	000E0	BEQL	7\$	
		04	0C	AC	C8	000E2	BISL2	SET_ATTRIB, 4(R7)	1436
				56	DD	000E7	PUSHL	R6	1437
	00000000G	00		01	FB	000E9	CALLS	#1, SYSSUPDATE	
		58		50	D0	000F0	MOVL	R0, STATUS	
				56	DD	000F3	PUSHL	R6	1443

RDBSHR
V04-000

RDBSHR - Rights database loadable system service 16-Sep-1984 01:48:50
SYSSMOD_IDENT_ATTRIB - Modify identifier att 14-Sep-1984 12:40:52

VAX-11 Bliss-32 V4.0-742
[LOADSS.SRC]RDBSHR.B32;1

Page 45
(8)

00000000G	00	01	FB	000F5	CALLS	#1, SYSS\$FREE	:	
35	A6	14	AE	90 000FC	MOVB	KRFS\$AV, 53(R6)	:	1448
34	A6	10	AE	90 00101	MOVB	KSZ\$AV, 52(R6)	:	1449
30	A6	0C	AE	D0 00106	MOVL	KBFS\$AV, 48(R6)	:	1450
	6A	08	AE	90 0010B	MOVB	RAC\$AV, (R10)	:	1451
04	A6	04	AE	D0 0010F	MOVL	ROPS\$AV, 4(R6)	:	1452
20	A6		6E	B0 00114	MOVW	USZ\$AV, 32(R6)	:	1453
	0C		58	E9 00118	BLBC	STATUS, 9\$:	1458
			56	DD 0011B	PUSHL	R6	:	1459
00000000G	00	01	FB	0011D	CALLS	#1, SYSS\$REWIND	:	
	58	50	D0	00124	MOVL	R0, STATUS	:	
	50	58	D0	00127 9\$:	MOVL	STATUS, R0	:	1462
		04	0012A		RET		:	1464

; Routine Size: 299 bytes, Routine Base: \$CODE\$ + 0A7D

; 1472 1465 1

```
1474 1466 1 %SBTTL ' SYSSMOD_IDENT_ID - Modify identifier value'
1475 1467 1 ROUTINE SYSSMOD_IDENT_ID ( RAB_PTR, ID, NEW_ID ) =
1476 1468 1
1477 1469 1 ++
1478 1470 1
1479 1471 1 FUNCTIONAL DESCRIPTION:
1480 1472 1
1481 1473 1 This routine modifies the name of the specified
1482 1474 1 identifier.
1483 1475 1
1484 1476 1 CALLING SEQUENCE:
1485 1477 1 SYSSMOD_IDENT_ID ( RAB_PTR, ID, .NEW_ID )
1486 1478 1
1487 1479 1 INPUT PARAMETERS:
1488 1480 1 RAB_PTR: Address of RAB for the open rights data base file
1489 1481 1 ID: identifier longword
1490 1482 1 NEW_ID: new value for identifier
1491 1483 1
1492 1484 1 IMPLICIT INPUTS:
1493 1485 1 NONE
1494 1486 1
1495 1487 1 OUTPUT PARAMETERS:
1496 1488 1 NONE
1497 1489 1
1498 1490 1 IMPLICIT OUTPUTS:
1499 1491 1 NONE
1500 1492 1
1501 1493 1 ROUTINE VALUE:
1502 1494 1 Status of operation
1503 1495 1
1504 1496 1 SIDE EFFECTS:
1505 1497 1 Identifier record modified
1506 1498 1
1507 1499 1 --
1508 1500 1
1509 1501 2 BEGIN
1510 1502 2
1511 1503 2
1512 1504 2 If the size of the holder ever changes then the OLD_HOLDER and NEW_HOLDER
1513 1505 2 vectors will have to be adjusted.
1514 1506 2
1515 1507 2 $ASSUME ( KGBSS_HOLDER, EQL, 8 ) ;
1516 1508 2
1517 1509 2 LABEL
1518 1510 2 MOD_ID ;
1519 1511 2
1520 1512 2 BIND
1521 1513 2 RAB = .RAB_PTR : $RAB_DECL ,
1522 1514 2 REC_BUFF = .RAB[RAB$L_UBF] : $BBLCK ;
1523 1515 2
1524 1516 2 LOCAL
1525 1517 2 KRFSAV : BYTE ,
1526 1518 2 KSZSAV : BYTE ,
1527 1519 2 KBFSAV : LONG ,
1528 1520 2 RACSAV : BYTE ,
1529 1521 2 ROPSAV : LONG ,
1530 1522 2 USZSAV : WORD ,
```



```
: 1531      1523      2      OLD_HOLDER : VECTOR [2, LONG],  
: 1532      1524      2      NEW_HOLDER : VECTOR [2, LONG],  
: 1533      1525      2      STATUS      : LONG ;  
: 1534      1526      2  
: 1535      1527      2      KRFSAV = .RAB[RAB$B_KRF] ;  
: 1536      1528      2      KSZSAV = .RAB[RAB$B_KSZ] ;  
: 1537      1529      2      KBFSAV = .RAB[RAB$L_KBF] ;  
: 1538      1530      2      RACSAV = .RAB[RAB$B_RAC] ;  
: 1539      1531      2      OPSAV = .RAB[RAB$L_ROP] ;  
: 1540      1532      2      USZSAV = .RAB[RAB$W_USZ] ;  
: 1541      1533      2  
: 1542      1534      2      MOD_ID:  
: 1543      1535      2      -BEGIN  
: 1544      1536      2  
: 1545      1537      2      |  
: 1546      1538      2      | Make sure that the new value is not in use.  
: 1547      1539      2      |  
: 1548      1540      2      RAB[RAB$B_RAC] = RAB$C_KEY ;  
: 1549      1541      2      RAB[RAB$B_KRF] = 0 ;  
: 1550      1542      2      RAB[RAB$B_KSZ] = 4 ;  
: 1551      1543      2      RAB[RAB$L_KBF] = NEW_ID ;  
: 1552      1544      2      RAB[RAB$W_USZ] = KGB$K_IDENT_RECORD ;  
: 1553      1545      2      RAB[RAB$L_ROP] = RAB$M_NLK OR RAB$M_RRL ; ! No lock, read regardless  
: 1554      1546      2      STATUS = $FIND ( RAB = RAB ) ;  
: 1555      1547      2      IF .STATUS THEN STATUS = SS$DUPLNAM ;  
: 1556      1548      2      IF .STATUS NEQ RMSS_RNF THEN LEAVE MOD_ID ;  
: 1557      1549      2  
: 1558      1550      2      |  
: 1559      1551      2      | Read the maintenance record to interlock the whole  
: 1560      1552      2      | operation.  
: 1561      1553      2      |  
: 1562      1554      2      RAB[RAB$L_KBF] = UPLIT (0) ;  
: 1563      1555      2      RAB[RAB$W_USZ] = KGB$K_MAINT_RECORD ;  
: 1564      1556      2      RAB[RAB$L_ROP] = RAB$M_WAT OR RAB$M_RLK OR RAB$M_ULK ;  
: 1565      1557      2      STATUS = $GET ( RAB = RAB ) ;  
: 1566      1558      2      IF NOT .STATUS THEN LEAVE MOD_ID ;  
: 1567      1559      2  
: 1568      1560      2      |  
: 1569      1561      2      | We will now loop through all the holder records and modify them  
: 1570      1562      2      | by reading in the ident record, change the value, delete the old record  
: 1571      1563      2      | record and write out the new one. The old one must be deleted  
: 1572      1564      2      | not updated because we are modifying the primary key.  
: 1573      1565      2      |  
: 1574      1566      2      RAB[RAB$L_KBF] = ID ;  
: 1575      1567      2      RAB[RAB$W_USZ] = KGB$K_IDENT_RECORD ;  
: 1576      1568      2      RAB[RAB$L_ROP] = RAB$M_LIM OR RAB$M_WAT OR RAB$M_RLK OR RAB$M_ULK ;  
: 1577      1569      2  
: 1578      1570      2      WHILE 1 DO  
: 1579      1571      4      BEGIN  
: 1580      1572      4      STATUS = $GET ( RAB = RAB ) ;  
: 1581      1573      4      IF ( .STATUS EQLU RMSS_EOF ) OR ( .STATUS EQLU RMSS_RNF ) THEN EXITLOOP ;  
: 1582      1574      4      IF NOT .STATUS THEN LEAVE MOD_ID ;  
: 1583      1575      4  
: 1584      1576      4      REC_BUFF[KGB$L_IDENTIFIER] = .NEW_ID ;  
: 1585      1577      4  
: 1586      1578      4      STATUS = $DELETE ( RAB = RAB ) ;  
: 1587      1579      4      IF NOT .STATUS THEN LEAVE MOD_ID ;
```

```
1588 1580 4
1589 1581 4      STATUS = $PUT ( RAB = RAB ) ;
1590 1582 4      IF NOT .STATUS THEN LEAVE MOD_ID ;
1591 1583 4
1592 1584 4      END ;
1593 1585 4
1594 1586 4
1595 1587 4      !
1596 1588 4      ! Now fix all the holder records
1597 1589 4      !
1598 1590 4      $REWIND ( RAB = RAB ) ;
1599 1591 4      OLD_HOLDER[0] = .ID ;
1600 1592 4      OLD_HOLDER[1] = 0 ;
1601 1593 4      NEW_HOLDER[0] = .NEW_ID ;
1602 1594 4      NEW_HOLDER[1] = 0 ;
1603 1595 4      RAB[RAB$B_KRF] = 1 ;
1604 1596 4      RAB[RAB$L_KBF] = OLD_HOLDER ;
1605 1597 4      RAB[RAB$B_KSZ] = KGB$S_HOLDER ;
1606 1598 4      RAB[RAB$W_USZ] = KGB$K_HOLD_RECORD ;
1607 1599 4      WHILE 1 DO
1608 1600 4      BEGIN
1609 1601 4      STATUS = $GET ( RAB = RAB ) ;
1610 1602 4      IF ( .STATUS EQLU RMSS$ EOF ) OR ( .STATUS EQLU RMSS$ _RNF ) THEN EXITLOOP ;
1611 1603 4      IF NOT .STATUS THEN LEAVE MOD_ID ;
1612 1604 4
1613 1605 4      CH$MOVE ( KGB$S_HOLDER, NEW_HOLDER, REC_BUFF[KGB$Q_HOLDER] ) ;
1614 1606 4
1615 1607 4      STATUS = $UPDATE ( RAB = RAB ) ;
1616 1608 4      IF NOT .STATUS THEN LEAVE MOD_ID ;
1617 1609 4
1618 1610 4      END ;
1619 1611 4
1620 1612 4      STATUS = SS$_NORMAL ;
1621 1613 4
1622 1614 4      END ;          ! End of MOD_ID
1623 1615 4
1624 1616 4      !
1625 1617 4      ! Clean up locks.
1626 1618 4      !
1627 1619 4      $FREE ( RAB = RAB ) ;
1628 1620 4
1629 1621 4      !
1630 1622 4      ! Restore RAB
1631 1623 4      !
1632 1624 4      RAB[RAB$B_KRF] = .KRFSAV ;
1633 1625 4      RAB[RAB$B_KSZ] = .KSZSAV ;
1634 1626 4      RAB[RAB$L_KBF] = .KBFSAV ;
1635 1627 4      RAB[RAB$B_RAC] = .RACSAV ;
1636 1628 4      RAB[RAB$L_ROP] = .ROPSAV ;
1637 1629 4      RAB[RAB$W_USZ] = .USZSAV ;
1638 1630 4
1639 1631 4      !
1640 1632 4      ! Get back to the beginning
1641 1633 4      !
1642 1634 4      IF .STATUS
1643 1635 4      THEN STATUS = $REWIND ( RAB = RAB ) ;
1644 1636 4
```


: 1645
: 1646
: 16471637 2 RETURN .STATUS ;
1638 2
1639 1 END ;

! End of SYSSMOD_IDENT_ID

.PSECT \$SPLITS\$,NOWRT,NOEXE,2

00000000 00176
00178 P.AAP:.BLKB 2
.LONG 0

.EXTRN SYSS\$DELETE

.PSECT \$CODE\$,NOWRT,2

OFFC 00000 SYSSMOD_IDENT_ID:

	5E		28	C2	00002	.WORD	Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11	: 1467
	56	04	AC	D0	00005	SUBL2	#40, SP	: 1513
	5B	24	A6	D0	00009	MOVL	RAB PTR, R6	: 1514
14	AE	35	A6	90	0000D	MOVL	36(R6), R11	: 1527
10	AE	34	A6	90	00012	MOVB	53(R6), KRFSAV	: 1528
	57	30	A6	9E	00017	MOVB	52(R6), KSZSAV	: 1529
0C	AE		67	D0	0001B	MOVAB	48(R6), R7	: 1530
08	AE	1E	A6	90	0001F	MOVL	(R7), KBFSAV	: 1531
	5A	04	A6	9E	00024	MOVB	30(R6), RACSAV	: 1532
04	AE		6A	D0	00028	MOVAB	4(R6), R10	: 1533
	59	20	A6	9E	0002C	MOVL	(R10), ROPSAV	: 1534
	6E		69	B0	00030	MOVAB	32(R6), R9	: 1535
1E	A6		01	90	00033	MOVW	(R9), USZSAV	: 1540
34	A6		04	B0	00037	MOVB	#1, 30(R6)	: 1542
	67	0C	AC	9E	0003B	MOVW	#4, 52(R6)	: 1543
	69		30	B0	0003F	MOVAB	NEW_ID, (R7)	: 1544
	6A	00100008	8F	D0	00042	MOVW	#48, (R9)	: 1545
			56	DD	00049	MOVL	#1048584, (R10)	: 1546
00000000G	00		01	FB	0004B	PUSHL	R6	: 1547
	58		50	D0	00052	CALLS	#1, SYSS\$FIND	: 1548
	04		58	E9	00055	MOVL	R0, STATUS	: 1554
	58	94	8F	9A	00058	BLBC	STATUS, 1\$: 1555
000182B2	8F		58	D1	0005C	MOVZBL	#148, STATUS	: 1556
			70	12	00063	CMPL	STATUS, #98994	: 1557
	67	0000'	CF	9E	00065	BNEQ	3\$: 1558
	69	40	8F	9B	0006A	MOVAB	P.AAP, (R7)	: 1566
	6A	000E0000	8F	D0	0006E	MOVZBW	#64, (R9)	: 1567
			56	DD	00075	MOVL	#917504, (R10)	: 1568
00000000G	00		01	FB	00077	PUSHL	R6	: 1572
	58		50	D0	0007E	CALLS	#1, SYSS\$GET	: 1573
	51		58	E9	00081	MOVL	R0, STATUS	: 1574
	67	08	AC	9E	00084	BLBC	STATUS, 3\$: 1575
	69		30	B0	00088	MOVAB	ID, (R7)	: 1576
	6A	000E4000	8F	D0	0008B	MOVW	#48, (R9)	: 1577
			56	DD	00092	MOVL	#933888, (R10)	: 1578
00000000G	00		01	FB	00094	PUSHL	R6	: 1579
	58		50	D0	0009B	CALLS	#1, SYSS\$GET	: 1580
0001827A	8F		58	D1	0009E	MOVL	R0, STATUS	: 1581
			30	13	000A5	CMPL	STATUS, #98938	: 1582
000182B2	8F		58	D1	000A7	BEQL	4\$: 1583
						CMPL	STATUS, #98994	: 1584

			27	13	000AE	BEQL	4\$		
	68		58	E9	000B0	BLBC	STATUS, 6\$	1574	
	6B	0C	AC	D0	000B3	MOVL	NEW_ID, (R11)	1576	
			56	DD	000B7	PUSHL	R6	1578	
00000000G	00		01	FB	000B9	CALLS	#1, SYSS\$DELETE		
	58		50	D0	000C0	MOVL	R0, STATUS		
	72		58	E9	000C3	BLBC	STATUS, 8\$	1579	
			56	DD	000C6	PUSHL	R6	1581	
00000000G	00		01	FB	000C8	CALLS	#1, SYSS\$PUT		
	58		50	D0	000CF	MOVL	R0, STATUS		
	BD		58	E8	000D2	BLBS	STATUS, 2\$	1582	
			61	11	000D5	BRB	8\$		
			56	DD	000D7	PUSHL	R6	1590	
00000000G	00		01	FB	000D9	CALLS	#1, SYSS\$REWIND		
	20	08	AC	D0	000E0	MOVL	ID, OLD HOLDER	1591	
		24	AE	D4	000E5	CLRL	OLD HOLDER+4	1592	
	18	0C	AC	D0	000E8	MOVL	NEW_ID, NEW HOLDER	1593	
		1C	AE	D4	000ED	CLRL	NEW HOLDER+4	1594	
	67	20	AE	9E	000F0	MOVAB	OLD HOLDER, (R7)	1596	
	34	0108	8F	B0	000F4	MOVW	#264, 52(R6)	1597	
	69		10	B0	000FA	MOVW	#16, (R9)	1598	
			56	DD	000FD	PUSHL	R6	1601	
00000000G	00		01	FB	000FF	CALLS	#1, SYSS\$GET		
	58		50	D0	00106	MOVL	R0, STATUS		
0001827A	8F		58	D1	00109	CMPL	STATUS, #98938	1602	
			23	13	00110	BEQL	7\$		
000182B2	8F		58	D1	00112	CMPL	STATUS, #98994		
			1A	13	00119	BEQL	7\$		
	1A		58	E9	0011B	BLBC	STATUS, 8\$	1603	
08 AB 18	AE		08	28	0011E	MOVC3	#8, NEW HOLDER, 8(R11)	1605	
			56	DD	00124	PUSHL	R6	1607	
00000000G	00		01	FB	00126	CALLS	#1, SYSS\$UPDATE		
	58		50	D0	0012D	MOVL	R0, STATUS		
	CA		58	E8	00130	BLBS	STATUS, 5\$	1608	
			03	11	00133	BRB	8\$		
	58		01	D0	00135	MOVL	#1, STATUS	1612	
			56	DD	00138	PUSHL	R6	1619	
00000000G	00		01	FB	0013A	CALLS	#1, SYSS\$FREE		
	35	14	AE	90	00141	MOVB	KRFS AV, 53(R6)	1624	
	34	10	AE	90	00146	MOVB	KSZ SAV, 52(R6)	1625	
	67	0C	AE	D0	0014B	MOVL	KBFS AV, (R7)	1626	
	1E	08	AE	90	0014F	MOVB	RACSAV, 30(R6)	1627	
	6A	04	AE	D0	00154	MOVL	ROPS AV, (R10)	1628	
	69		6E	B0	00158	MOVW	USZ SAV, (R9)	1629	
	0C		58	E9	0015B	BLBC	STATUS, 9\$	1634	
			56	DD	0015E	PUSHL	R6	1635	
00000000G	00		01	FB	00160	CALLS	#1, SYSS\$REWIND		
	58		50	D0	00167	MOVL	R0, STATUS		
	50		58	D0	0016A	MOVL	STATUS, R0	1637	
			04	0016D	RET			1639	

; Routine Size: 366 bytes, Routine Base: \$CODE\$ + 0BA8

; 1648

1640 1


```
: 1650      1641 1 %SBTTL ' SYSSMOD_IDENT_NAME - Modify identifier ame'
: 1651      1642 1 ROUTINE SYSSMOD_IDENT_NAME ( RAB_PTR, ID, NEW_NAMLEN, NEW_NAMADR) =
: 1652      1643 1
: 1653      1644 1 !++
: 1654      1645 1
: 1655      1646 1 FUNCTIONAL DESCRIPTION:
: 1656      1647 1
: 1657      1648 1     This routine modifies the name of the specified
: 1658      1649 1     identifier.
: 1659      1650 1
: 1660      1651 1 CALLING SEQUENCE:
: 1661      1652 1     SYSSMOD_IDENT_NAME ( RAB_PTR, ID, .NEW_NAMLEN, .NEW_NAMADR)
: 1662      1653 1
: 1663      1654 1 INPUT PARAMETERS:
: 1664      1655 1     RAB_PTR:      Address of RAB for the open rights data base file
: 1665      1656 1     ID:           identifier longword
: 1666      1657 1     NEW_NAMLEN:   Length of new name string
: 1667      1658 1     NEW_NAMADR:   Address of new name string
: 1668      1659 1
: 1669      1660 1 IMPLICIT INPUTS:
: 1670      1661 1     NONE
: 1671      1662 1
: 1672      1663 1 OUTPUT PARAMETERS:
: 1673      1664 1     NONE
: 1674      1665 1
: 1675      1666 1 IMPLICIT OUTPUTS:
: 1676      1667 1     NONE
: 1677      1668 1
: 1678      1669 1 ROUTINE VALUE:
: 1679      1670 1     Status of operation
: 1680      1671 1
: 1681      1672 1 SIDE EFFECTS:
: 1682      1673 1     Identifier record modified
: 1683      1674 1
: 1684      1675 1 --
: 1685      1676 1
: 1686      1677 2 BEGIN
: 1687      1678 2
: 1688      1679 2 LABEL
: 1689      1680 2     MOD_NAME ;
: 1690      1681 2
: 1691      1682 2 BIND
: 1692      1683 2     RAB      = .RAB_PTR      : $RAB_DECL ,
: 1693      1684 2     REC_BUFF = .RAB[RAB$L_UBF] : $BBLOCK ;
: 1694      1685 2
: 1695      1686 2 LOCAL
: 1696      1687 2     KRFSAV   : BYTE ,
: 1697      1688 2     KSZSAV   : BYTE ,
: 1698      1689 2     KBFSAV   : LONG ,
: 1699      1690 2     RACSAV   : BYTE ,
: 1700      1691 2     ROPSAV   : LONG ,
: 1701      1692 2     USZSAV   : WORD ,
: 1702      1693 2     NAME_BUFFER : $BBLOCK [KGB$S_NAME],
: 1703      1694 2     STATOS    : LONG ;
: 1704      1695 2
: 1705      1696 2 KRFSAV = .RAB[RAB$B_KRF] ;
: 1706      1697 2 KSZSAV = .RAB[RAB$B_KSZ] ;
```

```
1707 1698 2 KBFSAV = .RAB[RAB$K_KBF] ;
1708 1699 2 RACSAV = .RAB[RAB$K_RAC] ;
1709 1700 2 ROPSAV = .RAB[RAB$K_ROP] ;
1710 1701 2 USZSAV = .RAB[RAB$K_USZ] ;
1711 1702 2
1712 1703 2 MOD_NAME:
1713 1704 2 BEGIN
1714 1705 2
1715 1706 2 | First find out if there is a record with the new name already
1716 1707 2 |
1717 1708 2 | CH$TRANSLATE ( EXIST ID UPCASE, .NEW_NAMLEN, .NEW_NAMADR,
1718 1709 2 | RAB$K_NAME, NAME_BUFFER );
1719 1710 2 |
1720 1711 2 | RAB[RAB$K_KRF] = 2 ; | Id name key
1721 1712 2 | RAB[RAB$K_KSZ] = KGB$K_NAME ; | Key size
1722 1713 2 | RAB[RAB$K_KBF] = NAME_BUFFER ; | Name string address
1723 1714 2 | RAB[RAB$K_ROP] = RAB$K_NLK OR RAB$K_RRL ; | No lock, read regardless
1724 1715 2 | STATUS = $FIND ( RAB = RAB ) ;
1725 1716 2 | IF .STATUS THEN STATUS = SS$ DUPLNAM ;
1726 1717 2 | IF .STATUS NEQ RMSS_RNF THEN LEAVE MOD_NAME ;
1727 1718 2 |
1728 1719 2 | The name doesn't exist. Now we will get back to the beginning
1729 1720 2 | of the file and find the record that needs modification.
1730 1721 2 |
1731 1722 2 | STATUS = $REWIND ( RAB = RAB ) ;
1732 1723 2 | IF NOT .STATUS THEN LEAVE MOD_NAME ;
1733 1724 2 | RAB[RAB$K_KRF] = 0 ; | Id value key
1734 1725 2 | RAB[RAB$K_KSZ] = 4 ; | Key size
1735 1726 2 | RAB[RAB$K_KBF] = ID ; | ID value address
1736 1727 2 | RAB[RAB$K_ROP] = RAB$K_WAT OR | Wait if locked
1737 1728 2 | RAB$K_RLK OR | Lock record
1738 1729 2 | RAB$K_ULK ; | manual unlock
1739 1730 2 | RAB[RAB$K_USZ] = KGB$K_IDENT_RECORD ; | Ident record size
1740 1731 2 | STATUS = $GET ( RAB = RAB ) ;
1741 1732 2 | IF .STATUS EQL RMSS_RNF THEN STATUS = SS$ NOSUCHID ;
1742 1733 2 | IF NOT .STATUS THEN LEAVE MOD_NAME ;
1743 1734 2 |
1744 1735 2 | Move the new name into the record and update the file.
1745 1736 2 |
1746 1737 2 | CH$MOVE ( KGB$K_NAME, NAME_BUFFER, REC_BUFF[KGB$K_NAME] ) ;
1747 1738 2 | STATUS = $UPDATE ( RAB = RAB ) ;
1748 1739 2 |
1749 1740 2 | END ; | End of MOD_NAME
1750 1741 2
1751 1742 2 | Clean up locks.
1752 1743 2 |
1753 1744 2 | $FREE ( RAB = RAB ) ;
1754 1745 2 |
1755 1746 2 | Restore RAB
1756 1747 2 |
1757 1748 2 |
1758 1749 2 |
1759 1750 2 |
1760 1751 2 |
1761 1752 2 RAB[RAB$K_KRF] = .KRFSKV ;
1762 1753 2 RAB[RAB$K_KSZ] = .KSZSAV ;
1763 1754 2 RAB[RAB$K_KBF] = .KBFSKV ;
```



```
: 1764 1755 2 RAB[RAB$B_RAC] = .RACSAV ;
: 1765 1756 2 RAB[RAB$L_ROP] = .ROPSAV ;
: 1766 1757 2 RAB[RAB$W_USZ] = .USZSAV ;
: 1767 1758 2
: 1768 1759 2
: 1769 1760 2 ! Get back to the beginning
: 1770 1761 2
: 1771 1762 2 IF .STATUS
: 1772 1763 2 THEN STATUS = $REWIND ( RAB = RAB ) ;
: 1773 1764 2
: 1774 1765 2 RETURN .STATUS ;
: 1775 1766 2
: 1776 1767 1 END ;
```

! End of SYSSMOD_IDENT_NAME

				OFFC 00000 SYSSMOD_IDENT_NAME:		
			5E	28 C2 00002	.WORD Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11	: 1642
			56 04 AC D0 00005	SUBL2 #40, SP		: 1683
			58 24 A6 D0 00009	MOVL RAB_PTR, R6		: 1684
	04		AE 35 A6 90 0000D	MOVL 36(R6), R8		: 1696
			6E 34 A6 90 00012	MOVB 53(R6), KRFSAV		: 1697
				MOVB 52(R6), KSZSAV		: 1698
				PUSHL 48(R6)		: 1699
			5B 1E A6 90 00019	MOVB 30(R6), RACSAV		: 1700
			5A 04 A6 D0 0001D	MOVL 4(R6), ROPSAV		: 1701
			59 20 A6 B0 00021	MOVW 32(R6), USZSAV		: 1709
00000000G 00	20	10	BC 0C AC 2E 00025	MOVTC NEW NAMLEN, @NEW_NAMADR, #32, -		
		0C	AE 20 00030	EXEST_ID_UPCASE, #32, NAME_BUFFER		
		34	A6 0220 8F B0 00033	MOVW #544, 52(R6)		: 1712
		30	A6 0C AE 9E 00039	MOVAB NAME_BUFFER, 48(R6)		: 1713
		04	A6 00100008 8F D0 0003E	MOVL #1048584, 4(R6)		: 1714
				PUSHL R6		: 1715
		00000000G 00	01 FB 00048	CALLS #1, SYSS\$FIND		
		57	50 D0 0004F	MOVL R0, STATUS		
		04	57 E9 00052	BLBC STATUS, 1\$: 1716
		57	8F 9A 00055	MOVZBL #148, STATUS		
000182B2		8F	57 D1 00059 1\$:	CMPL STATUS, #98994		: 1717
			53 12 00060	BNEQ 3\$		
			56 DD 00062	PUSHL R6		: 1723
		00000000G 00	01 FB 00064	CALLS #1, SYSS\$REWIND		
		57	50 D0 0006B	MOVL R0, STATUS		
		44	57 E9 0006E	BLBC STATUS, 3\$: 1724
		34	A6 04 B0 00071	MOVW #4, 52(R6)		: 1726
		30	A6 08 AC 9E 00075	MOVAB ID, 48(R6)		: 1727
		04	A6 000E0000 8F D0 0007A	MOVL #917504, 4(R6)		: 1729
		20	A6 30 B0 00082	MOVW #48, 32(R6)		: 1731
			56 DD 00086	PUSHL R6		: 1732
		00000000G 00	01 FB 00088	CALLS #1, SYSS\$GET		
		57	50 D0 0008F	MOVL R0, STATUS		
000182B2		8F	57 D1 00092	CMPL STATUS, #98994		: 1733
			05 12 00099	BNEQ 2\$		
		57	8F 3C 0009B	MOVZWL #8684, STATUS		
		12	57 E9 000A0 2\$:	BLBC STATUS, 3\$: 1734
10 A8 0C		AE	20 28 000A3	MOVW #32, NAME_BUFFER, 16(R8)		: 1739

RDBSHR
V04-000

RDBSHR - Rights database loadable system service
SYSSMOD_IDENT_NAME - Modify identifier ame

K 11

16-Sep-1984 01:48:50
14-Sep-1984 12:40:52

VAX-11 Bliss-32 V4.0-742
[LOADSS.SRC]RDBSHR.B32;1

Page 54
(10)

00000000G	00		56	DD	000A9	PUSHL	R6	:	1740
	57		01	FB	000AB	CALLS	#1, SYSSUPDATE	:	
			50	D0	000B2	MOVL	R0, STATUS	:	
			56	DD	000B5	PUSHL	R6	:	1747
00000000G	00		01	FB	000B7	CALLS	#1, SYSSFREE	:	
35	A6	08	AE	90	000BE	MOVB	KRFSAV, 53(R6)	:	1752
34	A6	04	AE	90	000C3	MOVB	KSZSAV, 52(R6)	:	1753
30	A6		6E	D0	000C8	MOVL	KBFSAV, 48(R6)	:	1754
1E	A6		5B	90	000CC	MOVB	RACSAV, 30(R6)	:	1755
04	A6		5A	D0	000D0	MOVL	ROPSAV, 4(R6)	:	1756
20	A6		59	B0	000D4	MOVW	USZSAV, 32(R6)	:	1757
	0C		57	E9	000D8	BLBC	STATUS, 4\$:	1762
			56	DD	000DB	PUSHL	R6	:	1763
00000000G	00		01	FB	000DD	CALLS	#1, SYSSREWIND	:	
	57		50	D0	000E4	MOVL	R0, STATUS	:	
	50		57	D0	000E7	MOVL	STATUS, R0	:	1765
			04	000EA	4\$:	RET		:	1767

; Routine Size: 235 bytes, Routine Base: \$CODE\$ + 0D16

; 1777 1768 1


```
: 1779 1769 1 %SBTTL ' SYS$REM_HOLDER - remove holder record'
: 1780 1770 1 GLOBAL ROUTINE SYS$REM_HOLDER (ID, HOLDER) =
: 1781 1771 1
: 1782 1772 1 !++
: 1783 1773 1
: 1784 1774 1 FUNCTIONAL DESCRIPTION:
: 1785 1775 1
: 1786 1776 1 This routine removes the specified holder record.
: 1787 1777 1
: 1788 1778 1 CALLING SEQUENCE:
: 1789 1779 1 SYS$REM_HOLDER (ID, HOLDER)
: 1790 1780 1
: 1791 1781 1 INPUT PARAMETERS:
: 1792 1782 1 ID: identifier longword
: 1793 1783 1 HOLDER: address of the holder identifier quadword
: 1794 1784 1
: 1795 1785 1 IMPLICIT INPUTS:
: 1796 1786 1 NONE
: 1797 1787 1
: 1798 1788 1 OUTPUT PARAMETERS:
: 1799 1789 1 NONE
: 1800 1790 1
: 1801 1791 1 IMPLICIT OUTPUTS:
: 1802 1792 1 NONE
: 1803 1793 1
: 1804 1794 1 ROUTINE VALUE:
: 1805 1795 1 Status of operation
: 1806 1796 1
: 1807 1797 1 SIDE EFFECTS:
: 1808 1798 1 Holder record removed
: 1809 1799 1
: 1810 1800 1 !--
: 1811 1801 1
: 1812 1802 2 BEGIN
: 1813 1803 2
: 1814 1804 2 LOCAL
: 1815 1805 2 LOC_ID : LONG, ! local copy of ID
: 1816 1806 2 LOC_HOLDER : REF VECTOR, ! local copy of HOLDER
: 1817 1807 2 HOLDER_ID : VECTOR [2], ! local copy of holder id quadword
: 1818 1808 2 STATUS : LONG, ! general status value
: 1819 1809 2 CLOSE : LONG, ! call to EXE$CLOSE_RDB required flag
: 1820 1810 2 RAB : $RAB_DECL, ! RAB for file operations
: 1821 1811 2 REC_BUFFER : $BBLOCK [KGB$K_IDENT_RECORD]; ! buffer to read records
: 1822 1812 2
: 1823 1813 2
: 1824 1814 2 LABEL
: 1825 1815 2 RDB_OPEN; ! rights database is open in this block
: 1826 1816 2
: 1827 1817 2 ! Validate parameters
: 1828 1818 2 !
: 1829 1819 2 !
: 1830 1820 2
: 1831 1821 2 LOC_ID = .ID;
: 1832 1822 2 IF (.LOC_ID AND UIC$M_ID_FORM_FLAG) NEQU 0
: 1833 1823 2 THEN
: 1834 1824 3 (IF (.LOC_ID GTRU UIC$K_LAST_ID) THEN RETURN SS$_IVIDENT)
: 1835 1825 2 ELSE
```

```
1836      2      (IF (.LOC_ID GTRU UIC$K_MAX_UIC) OR (.LOC_ID EQL 0) THEN RETURN SS$_IVIDENT);
1837
1838      2      LOC HOLDER = .HOLDER;
1839      2      IF NOT PROBER (%REF(0), %REF(8), .LOC_HOLDER) THEN RETURN SS$_ACCVIO;
1840      2      HOLDER_ID[0] = .LOC_HOLDER[0];
1841      2      HOLDER_ID[1] = .LOC_HOLDER[1];
1842      2      IF .HOLDER_ID[0] GTRU UIC$K_MAX_UIC OR .HOLDER_ID[1] NEQU 0
1843      2      THEN
1844      2      RETURN SS$_IVIDENT;
1845
1846      2      ! Get the rights database open for write.
1847      2      !
1848
1849      P 1839 2      $RAB_INIT (RAB = RAB,
1850      P 1840 2      RAC = KEY,
1851      P 1841 2      KRF = 0,
1852      P 1842 2      KBF = LOC_ID,
1853      P 1843 2      KSZ = 4,
1854      P 1844 2      ROP = (LIM, WAT, RLK, ULK),
1855      P 1845 2      UBF = REC_BUFFER,
1856      P 1846 2      USZ = KGB$K_IDENT_RECORD
1857      2      );
1858      2      STATUS = EXE$OPEN_RDB (0, 1, RAB[RAB$W_ISI], CLOSE);
1859      2      IF NOT .STATUS THEN RETURN .STATUS;
1860
1861      2      RDB_OPEN:
1862      2      BEGIN
1863      3      ! Read and lock the ident record.
1864      3      !
1865      3      STATUS = $GET (RAB = RAB);
1866      3      IF .STATUS EQLU RMSS$_RNF THEN STATUS = SS$_NOSUCHID;
1867      3      IF NOT .STATUS
1868      3      THEN
1869      4      BEGIN
1870      4      $FREE (RAB = RAB);
1871      4      LEAVE RDB_OPEN;
1872      4      END;
1873      3      ! Read the holder records looking for the specified one.
1874      3      !
1875      3      RAB[RAB$V_ULK] = 0;
1876      3      RAB[RAB$B_RAC] = RAB$C_SEQ;
1877      3      WHILE 1 DO
1878      4      BEGIN
1879      4      STATUS = $GET (RAB = RAB);
1880      4      IF .STATUS EQLU RMSS$_EOF OR .STATUS EQLU RMSS$_OK_LIM
1881      4      THEN
1882      5      BEGIN
1883      5      $FREE (RAB = RAB);
1884      5      STATUS = SS$_NOSUCHID;
1885      5      LEAVE RDB_OPEN;
1886      5      END;
1887      4      IF CH$EQL (KGB$$_HOLDER, HOLDER_ID[0], KGB$$_HOLDER, REC_BUFFER[KGB$Q_HOLDER])
```


RDBSHR
V04-000

RDBSHR - Rights database loadable system servic
SYSSREM_HOLDER - remove holder record

N 11
16-Sep-1984 01:48:50
14-Sep-1984 12:40:52

VAX-11 Bliss-32 V4.0-742
[LOADSS.SRC]RDBSHR.B32;1

Page 57
(11)

SYS
V04

```
1893 1883 4      THEN
1894 1884 4      EXITLOOP;
1895 1885 4      END;
1896 1886 4      ! Delete the located record.
1897 1887 4      !
1898 1888 4      !
1899 1889 4      !
1900 1890 4      STATUS = $DELETE (RAB = RAB);
1901 1891 4      $FREE (RAB = RAB);
1902 1892 4      END;
1903 1893 4      !
1904 1894 4      ! Close the rights database if there is no image
1905 1895 4      !
1906 1896 4      !
1907 1897 4      IF .CLOSE THEN EXE$CLOSE_RDB();
1908 1898 4      IF .STATUS
1909 1899 4      THEN
1910 1900 4      RETURN SS$_NORMAL
1911 1901 4      ELSE
1912 1902 4      RETURN .STATUS;
1913 1903 4      !
1914 1904 1  END;      ! End of routine SYSSREM_HOLDER
```

				00FC 00000	.ENTRY	SYSSREM HOLDER, Save R2,R3,R4,R5,R6,R7	1770
	57	00000000G	00	9E 00002	MOVAB	SYSS\$FREE, R7	
	56	00000000G	00	9E 00009	MOVAB	SYSS\$GET, R6	
	5E	80	AE	9E 00010	MOVAB	-128(SP), SP	
		04	AC	DD 00014	PUSHL	ID	1821
			0B	18 00017	BGEQ	1\$	1822
8FFFFFFF	8F		6E	D1 00019	CMPL	LOC_ID, #-1879048193	1824
			0F	1B 00020	BLEQU	2\$	
			33	11 00022	BRB	4\$	
3FFFFFFF	8F		6E	D1 00024	CMPL	LOC_ID, #1073741823	1826
			2A	1A 0002B	BGTRU	4\$	
			6E	D5 0002D	TSTL	LOC_ID	
			26	13 0002F	BEQL	4\$	
	50	08	AC	D0 00031	MOVL	HOLDER, LOC HOLDER	1828
60	08		00	0C 00035	PROBER	#0, #8, (LOC_HOLDER)	1829
			04	12 00039	BNEQ	3\$	
	50		0C	D0 0003B	MOVL	#12, R0	
			04	0003E	RET		
7C	AE		60	D0 0003F	MOVL	(LOC_HOLDER), HOLDER_ID	1830
FC	AD	04	A0	D0 00043	MOVL	4(LOC_HOLDER), HOLDER_ID+4	1831
3FFFFFFF	8F	7C	AE	D1 00048	CMPL	HOLDER_ID, #1073741823	1832
		FC	05	1A 00050	BGTRU	4\$	
			AD	D5 00052	TSTL	HOLDER_ID+4	
			06	13 00055	BEQL	5\$	
	50	2224	8F	3C 00057	MOVZWL	#8740, R0	1834
			04	0005C	RET		
0044	8F	00	6E	2C 0005D	MOVCS	#0, (SP), #0, #68, \$RMS_PTR	1847
			00	2C 0005D			
		38	AE	4401	MOVW	#17409, \$RMS_PTR	
	3C	AE	000E4000	8F	BO	00066	
			8F	DO	0006C	MOVL	#933888, \$RMS_PTR+4

56	AE	01	90	00074	MOVB	#1, \$RMS_PTR+30	:
58	AE	30	B0	00078	MOVW	#48, \$RMS_PTR+32	:
5C	AE	08	AE	9E 0007C	MOVAB	REC_BUFFER, \$RMS_PTR+36	:
68	AE	6E	9E	00081	MOVAB	LOC_ID, \$RMS_PTR+48	:
6C	AE	04	90	00085	MOVB	#4, \$RMS_PTR+52	:
		04	AE	9F 00089	PUSHAB	CLOSE	1848
		3E	AE	9F 0008C	PUSHAB	RAB+2	:
		01	DD	0008F	PUSHL	#1	:
		7E	D4	00091	CLRL	-(SP)	:
00000000G	9F	04	FB	00093	CALLS	#4, @#EXE\$OPEN_RDB	:
54		50	D0	0009A	MOVL	R0, STATUS	:
76		54	E9	0009D	BLBC	STATUS, 13\$	1849
		38	AE	9F 000A0	PUSHAB	RAB	1857
	66	01	FB	000A3	CALLS	#1, SYSS\$GET	:
	54	50	D0	000A6	MOVL	R0, STATUS	:
000182B2	8F	54	D1	000A9	CMPL	STATUS, #98994	1858
		05	12	000B0	BNEQ	6\$:
	54	21EC	8F	3C 000B2	MOVZWL	#8684, STATUS	:
	44		54	E9 000B7	BLBC	STATUS, 10\$	1859
3E	AE	04	8A	000BA	BICB2	#4, RAB+6	1869
		56	AE	94 000BE	CLRB	RAB+30	1870
		38	AE	9F 000C1	PUSHAB	RAB	1873
	66	01	FB	000C4	CALLS	#1, SYSS\$GET	:
	54	50	D0	000C7	MOVL	R0, STATUS	:
0001827A	8F	54	D1	000CA	CMPL	STATUS, #98938	1874
		09	13	000D1	BEQL	8\$:
00018051	8F	54	D1	000D3	CMPL	STATUS, #98385	:
		0D	12	000DA	BNEQ	9\$:
		38	AE	9F 000DC	PUSHAB	RAB	1877
	67	01	FB	000DF	CALLS	#1, SYSS\$FREE	:
	54	21EC	8F	3C 000E2	MOVZWL	#8684, STATUS	1878
		1B	11	000E7	BRB	11\$	1879
10	AE	7C	AE	08 29 000E9	CMPC3	#8, HOLDER_ID, REC_BUFFER+8	1882
			D0	12 000EF	BNEQ	7\$:
		38	AE	9F 000F1	PUSHAB	RAB	1890
00000000G	00	01	FB	000F4	CALLS	#1, SYSS\$DELETE	:
	54	50	D0	000FB	MOVL	R0, STATUS	:
		38	AE	9F 000FE	PUSHAB	RAB	1891
	67	01	FB	00101	CALLS	#1, SYSS\$FREE	:
	07	04	AE	E9 00104	BLBC	CLOSE, 12\$	1897
00000000G	9F	00	FB	00108	CALLS	#0, @#EXE\$CLOSE_RDB	:
	04	54	E9	0010F	BLBC	STATUS, 13\$	1898
	50	01	D0	00112	MOVL	#1, R0	1902
			04	00115	RET		:
	50	54	D0	00116	MOVL	STATUS, R0	1904
			04	00119	RET		:

; Routine Size: 282 bytes, Routine Base: \$CODE\$ + 0E01


```
: 1916 1905 1 %SBTTL ' SYSSREM_IDENT - remove identifier from RDB'
: 1917 1906 1 GLOBAL ROUTINE SYSSREM_IDENT (ID) =
: 1918 1907 1
: 1919 1908 1 ++
: 1920 1909 1
: 1921 1910 1 FUNCTIONAL DESCRIPTION:
: 1922 1911 1
: 1923 1912 1 This routine removes the specified identifier from the rights
: 1924 1913 1 database.
: 1925 1914 1
: 1926 1915 1 CALLING SEQUENCE:
: 1927 1916 1 SYSSREM_IDENT (ID)
: 1928 1917 1
: 1929 1918 1 INPUT PARAMETERS:
: 1930 1919 1 ID: identifier longword
: 1931 1920 1
: 1932 1921 1 IMPLICIT INPUTS:
: 1933 1922 1 NONE
: 1934 1923 1
: 1935 1924 1 OUTPUT PARAMETERS:
: 1936 1925 1 NONE
: 1937 1926 1
: 1938 1927 1 IMPLICIT OUTPUTS:
: 1939 1928 1 NONE
: 1940 1929 1
: 1941 1930 1 ROUTINE VALUE:
: 1942 1931 1 Status of operation
: 1943 1932 1
: 1944 1933 1 SIDE EFFECTS:
: 1945 1934 1 Identifier record removed
: 1946 1935 1
: 1947 1936 1 --
: 1948 1937 1
: 1949 1938 2 BEGIN
: 1950 1939 2
: 1951 1940 2 LOCAL
: 1952 1941 2 LOC_ID : VECTOR [2] INITIAL (0,0),
: 1953 1942 2 : local copy of ID
: 1954 1943 2 STATUS : LONG, : general status value
: 1955 1944 2 CLOSE : LONG, : call to EXE$CLOSE_RDB required flag
: 1956 1945 2 RAB : $RAB DECL, : RAB for file I/O
: 1957 1946 2 IDENT_RFA : $BBLOCK [RAB$$ RFA],
: 1958 1947 2 : RFA of ident record
: 1959 1948 2 REC_BUFFER : $BBLOCK [KGB$K IDENT_RECORD];
: 1960 1949 2 : Record buffer
: 1961 1950 2
: 1962 1951 2 LABEL
: 1963 1952 2 RDB_OPEN; ! rights database is open in this block
: 1964 1953 2
: 1965 1954 2 ! Validate ID
: 1966 1955 2 !
: 1967 1956 2
: 1968 1957 2 LOC_ID[0] = ID;
: 1969 1958 2 IF (.LOC_ID[0] AND UIC$M_ID_FORM_FLAG) NEQU 0
: 1970 1959 2 THEN
: 1971 1960 3 (IF (.LOC_ID[0] GTRU UIC$K_LAST_ID) THEN RETURN SS$_IVIDENT)
: 1972 1961 2 ELSE
```

```
: 1973      1962      2      (IF (.LOC_ID[0] GTRU UIC$K_MAX_UIC) OR (.LOC_ID[0] EQL 0) THEN RETURN SS$_IVIDENT);
: 1974      1963      2
: 1975      1964      2      ! Open the rights database for writing.
: 1976      1965      2      !
: 1977      1966      2
: 1978      P 1967      2      $RAB_INIT (RAB = RAB,
: 1979      P 1968      2              RAC = KEY,
: 1980      P 1969      2              KRF = 1,
: 1981      P 1970      2              KSZ = KGB$$ HOLDER,
: 1982      P 1971      2              KBF = LOC_ID[0],
: 1983      P 1972      2              USZ = KGB$K_IDENT_RECORD,
: 1984      P 1973      2              UBF = REC_BUFFER,
: 1985      P 1974      2              ROP = (LIM, WAT, RLK, ULK)
: 1986      1975      2              );
: 1987      1976      2      STATUS = EXE$OPEN RDB (0, 1, RAB[RAB$W_ISI], CLOSE);
: 1988      1977      2      IF NOT .STATUS THEN RETURN .STATUS;
: 1989      1978      2
: 1990      1979      2      RDB_OPEN:
: 1991      1980      2      BEGIN
: 1992      1981      3
: 1993      1982      3          ! Delete holder records held by this id
: 1994      1983      3          !
: 1995      1984      3
: 1996      1985      3          STATUS = $GET (RAB = RAB);
: 1997      1986      3          IF NOT .STATUS AND .STATUS NEQU RMSS$_RNF THEN LEAVE RDB_OPEN;
: 1998      1987      3          IF .STATUS
: 1999      1988      3          THEN
: 2000      1989      4              BEGIN
: 2001      1990      4              RAB[RAB$B_RAC] = RAB$C_SEQ;
: 2002      1991      4              WHILE 1 DO
: 2003      1992      5                  BEGIN
: 2004      1993      5                      STATUS = $DELETE (RAB = RAB);
: 2005      1994      5                      IF NOT .STATUS
: 2006      1995      5                      THEN
: 2007      1996      6                          BEGIN
: 2008      1997      6                              $FREE (RAB = RAB);
: 2009      1998      6                              LEAVE RDB_OPEN;
: 2010      1999      5                              END;
: 2011      2000      5                      STATUS = $FIND (RAB = RAB);
: 2012      2001      5                      IF .STATUS EQLU RMSS$_EOF OR .STATUS EQLU RMSS$_OK_LIM
: 2013      2002      5                      THEN
: 2014      2003      5                          EXITLOOP;
: 2015      2004      5                      IF NOT .STATUS
: 2016      2005      5                      THEN
: 2017      2006      6                          BEGIN
: 2018      2007      6                              $FREE (RAB = RAB);
: 2019      2008      6                              LEAVE RDB_OPEN;
: 2020      2009      5                              END;
: 2021      2010      4                      END;
: 2022      2011      3              END;
: 2023      2012      3
: 2024      2013      3          ! Now delete all holders of this identifier
: 2025      2014      3          !
: 2026      2015      3
: 2027      2016      3          RAB[RAB$B_RAC] = RAB$C_KEY;
: 2028      2017      3          RAB[RAB$B_KRF] = 0;
: 2029      2018      3          RAB[RAB$B_KSZ] = 4;
```



```
2030 2019 3
2031 2020 3
2032 2021 3
2033 2022 3
2034 2023 3
2035 2024 3
2036 2025 3
2037 2026 3
2038 2027 4
2039 2028 4
2040 2029 4
2041 2030 3
2042 2031 3
2043 2032 3
2044 2033 3
2045 2034 3
2046 2035 3
2047 2036 3
2048 2037 3
2049 2038 3
2050 2039 4
2051 2040 4
2052 2041 4
2053 2042 4
2054 2043 4
2055 2044 4
2056 2045 4
2057 2046 5
2058 2047 5
2059 2048 5
2060 2049 4
2061 2050 4
2062 2051 4
2063 2052 4
2064 2053 4
2065 2054 5
2066 2055 5
2067 2056 5
2068 2057 4
2069 2058 3
2070 2059 3
2071 2060 3
2072 2061 3
2073 2062 3
2074 2063 3
2075 2064 3
2076 2065 3
2077 2066 3
2078 2067 3
2079 2068 4
2080 2069 4
2081 2070 4
2082 2071 3
2083 2072 3
2084 2073 3
2085 2074 3
2086 2075 2

! First locate and lock the identifier record.
!
STATUS = $GET (RAB = RAB);
IF .STATUS EQLU RMSS_RNF THEN STATUS = SS$_NOSUCHID;
IF NOT .STATUS
THEN
    BEGIN
        $FREE (RAB = RAB);
        LEAVE RDB_OPEN;
    END;
CH$MOVE (RAB$_RFA, RAB[RAB$_RFA], IDENT_RFA);

! Now sequentially locate all the holder records and delete them.
!
RAB[RAB$_RAC] = RAB$_SEQ;
RAB[RAB$_ULK] = 0;
WHILE 1 DO
    BEGIN
        STATUS = $FIND (RAB = RAB);
        IF .STATUS EQLU RMSS_EOF OR .STATUS EQLU RMSS_OK_LIM
        THEN
            EXITLOOP;
        IF NOT .STATUS
        THEN
            BEGIN
                $FREE (RAB = RAB);
                LEAVE RDB_OPEN;
            END;

        STATUS = $DELETE (RAB = RAB);
        IF NOT .STATUS
        THEN
            BEGIN
                $FREE (RAB = RAB);
                LEAVE RDB_OPEN;
            END;
        END;

! Finally, re-locate and delete the identifier record.
!
RAB[RAB$_RAC] = RAB$_RFA;
CH$MOVE (RAB$_RFA, IDENT_RFA, RAB[RAB$_RFA]);
STATUS = $FIND (RAB = RAB);
IF NOT .STATUS
THEN
    BEGIN
        $FREE (RAB = RAB);
        LEAVE RDB_OPEN;
    END;

STATUS = $DELETE (RAB = RAB);
$FREE (RAB = RAB);
END;
```

```
2087  
2088  
2089  
2090  
2091  
2092  
2093  
2094  
2095  
2096  
2097  
2098  
2076 2  
2077 2  
2078 2  
2079 2  
2080 2 IF .CLOSE THEN EXE$CLOSE_RDB();  
2081 2 IF .STATUS  
2082 2 THEN  
2083 2 RETURN SS$_NORMAL  
2084 2 ELSE  
2085 2 RETURN .STATUS;  
2086 2  
2087 1 END;
```

! End of routine SYSS\$REM_IDENT

0044	8F	00	6E	3C	AE	4401	8F	B0	00050	MOVW	#17409, \$RMS_PTR	1906
				40	AE	000E4000	8F	D0	00056	MOVL	#933888, \$RMS_PTR+4	
				5A	AE		01	90	0005E	MOVB	#1, \$RMS_PTR+30	
				5C	AE		30	B0	00062	MOVW	#48, \$RMS_PTR+32	
				60	AE	04	AE	9E	00066	MOVAB	REC_BUFFER, \$RMS_PTR+36	
				6C	AE	F8	AD	9E	0006B	MOVAB	LOC_ID, \$RMS_PTR+48	
				70	AE	0108	8F	B0	00070	MOVW	#264, \$RMS_PTR+52	
						42	5E	DD	00076	PUSHL	SP	1976
							AE	9F	00078	PUSHAB	RAB+2	
							01	DD	0007B	PUSHL	#1	
							7E	D4	0007D	CLRL	-(SP)	
							04	FB	0007F	CALLS	#4, @EXE\$OPEN_RDB	
							50	D0	00086	MOVL	R0, STATUS	
							56	E8	00089	BLBS	STATUS, 4\$	1977
							00DF	31	0008C	BRW	16\$	
							AE	9F	0008F	PUSHAB	RAB	1985
							01	FB	00092	CALLS	#1, SYSS\$GET	
							50	D0	00095	MOVL	R0, STATUS	
							56	E8	00098	BLBS	STATUS, 6\$	1986
							56	D1	0009B	CMPL	STATUS, #98994	
							03	13	000A2	BEQL	5\$	

				00B6	31	000A4	BRW	14\$		
	2F			56	E9	000A7	5\$:	BLBC	STATUS, 8\$	1987
		5A		AE	94	000AA	6\$:	CLRB	RAB+30	1990
		3C		AE	9F	000AD	7\$:	PUSHAB	RAB	1993
	67			01	FB	000B0		CALLS	#1, SYSS\$DELETE	
	56			50	D0	000B3		MOVL	R0, STATUS	
	79			56	E9	000B6		BLBC	STATUS, 11\$	1994
		3C		AE	9F	000B9		PUSHAB	RAB	2000
	68			01	FB	000BC		CALLS	#1, SYSS\$FIND	
	56			50	D0	000BF		MOVL	R0, STATUS	
0001827A	8F			56	D1	000C2		CMPL	STATUS, #98938	2001
				0E	13	000C9		BEQL	8\$	
00018051	8F			56	D1	000CB		CMPL	STATUS, #98385	
				05	13	000D2		BEQL	8\$	
	D6			56	E8	000D4		BLBS	STATUS, 7\$	2004
				7A	11	000D7		BRB	13\$	2007
	5A	AE		01	90	000D9	8\$:	MOVB	#1, RAB+30	2016
	70	AE		04	B0	000DD		MOVW	#4, RAB+52	2018
			3C	AE	9F	000E1		PUSHAB	RAB	2023
	69			01	FB	000E4		CALLS	#1, SYSS\$GET	
	56			50	D0	000E7		MOVL	R0, STATUS	
000182B2	8F			56	D1	000EA		CMPL	STATUS, #98994	2024
				05	12	000F1		BNEQ	9\$	
	56		21EC	8F	3C	000F3		MOVZWL	#8684, STATUS	
	58			56	E9	000F8	9\$:	BLBC	STATUS, 13\$	2025
34	AE	4C		06	28	000FB		MOVC3	#6, RAB+16, IDENT_RFA	2031
			5A	AE	94	00101		CLRB	RAB+30	2036
	42	AE		04	8A	00104		BICB2	#4, RAB+6	2037
			3C	AE	9F	00108	10\$:	PUSHAB	RAB	2040
	68			01	FB	0010B		CALLS	#1, SYSS\$FIND	
	56			50	D0	0010E		MOVL	R0, STATUS	
0001827A	8F			56	D1	00111		CMPL	STATUS, #98938	2041
				1A	13	00118		BEQL	12\$	
00018051	8F			56	D1	0011A		CMPL	STATUS, #98385	
				11	13	00121		BEQL	12\$	
	2D			56	E9	00123		BLBC	STATUS, 13\$	2044
			3C	AE	9F	00126		PUSHAB	RAB	2051
	67			01	FB	00129		CALLS	#1, SYSS\$DELETE	
	56			50	D0	0012C		MOVL	R0, STATUS	
	D6			56	E8	0012F		BLBS	STATUS, 10\$	2052
				1F	11	00132	11\$:	BRB	13\$	2055
		5A	AE	02	90	00134	12\$:	MOVB	#2, RAB+30	2063
4C	AE	34	AE	06	28	00138		MOVC3	#6, IDENT_RFA, RAB+16	2064
			3C	AE	9F	0013E		PUSHAB	RAB	2065
	68			01	FB	00141		CALLS	#1, SYSS\$FIND	
	56			50	D0	00144		MOVL	R0, STATUS	
	09			56	E9	00147		BLBC	STATUS, 13\$	2066
			3C	AE	9F	0014A		PUSHAB	RAB	2073
	67			01	FB	0014D		CALLS	#1, SYSS\$DELETE	
	56			50	D0	00150		MOVL	R0, STATUS	
			3C	AE	9F	00153	13\$:	PUSHAB	RAB	2074
00000000G	00			01	FB	00156		CALLS	#1, SYSS\$FREE	
	07			6E	E9	0015D	14\$:	BLBC	CLOSE, 15\$	2080
00000000G	9F			00	FB	00160		CALLS	#0, @#EXE\$CLOSE_RDB	
	04			56	E9	00167	15\$:	BLBC	STATUS, 16\$	2081
	50			01	D0	0016A		MOVL	#1, R0	2085
				04	00	0016D		RET		

```

; Routine Size: 370 bytes,      Routine Base: $CODE$ + 0F1B

```

```

: 2099      2088 1
: 2100      2089 1 END
: 2101      2090 0 ELUDOM

```

PSECT SUMMARY

Name	Bytes	Attributes
\$CODE\$	4237	NOVEC,NOWRT, RD , EXE,NOSHR, LCL, REL, CON,NOPIC,ALIGN(2)
\$SPLITS	380	NOVEC,NOWRT, RD ,NOEXE,NOSHR, LCL, REL, CON,NOPIC,ALIGN(2)

Library Statistics

File	----- Total	Symbols Loaded	----- Percent	Pages Mapped	Processing Time
_\$255\$DUA28:[SYSLIB]LIB.L32;1	18619	195	1	1000	00:01.9

COMMAND QUALIFIERS

```

; BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS$:RDBSHR/OBJ=OBJ$:RDBSHR MSRC$:RDBSHR/UPDATE=(ENH$:RDBSHR)

```

```

; Size:      4237 code + 380 data bytes
; Run Time:   01:33.8
; Elapsed Time: 02:52.5
; Lines/CPU Min: 1337
; Lexemes/CPU-Min: 30273
; Memory Used: 308 pages
; Compilation Complete

```


0220

AH-BT13A-SE
 VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY